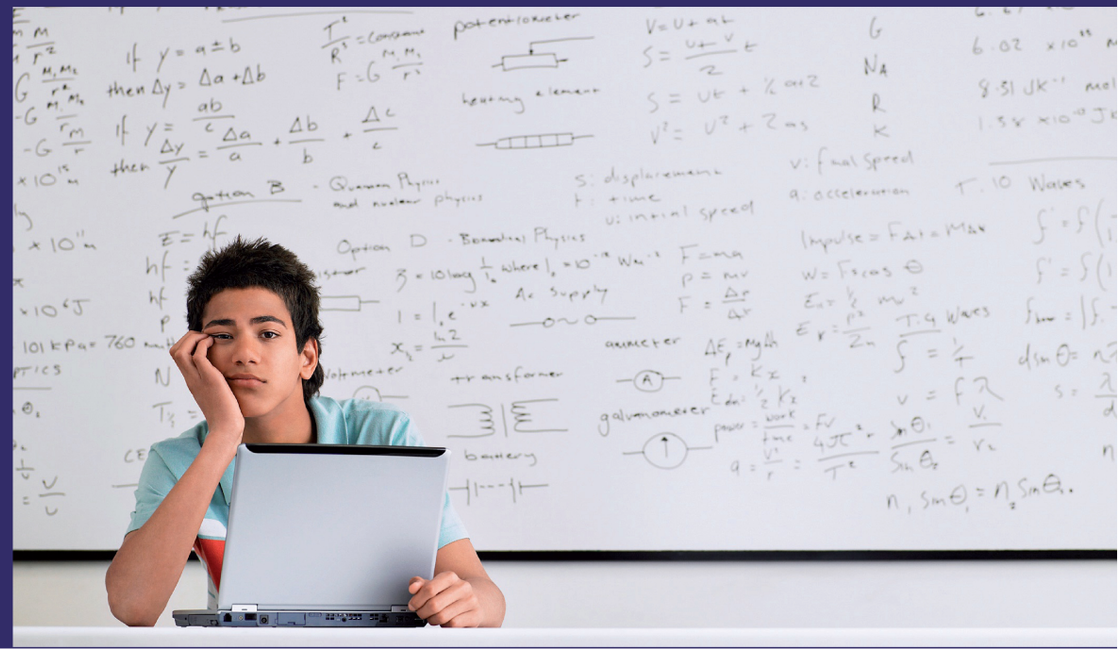
В книге рассмотрены современные методы распознавания образов: классическая теория принятия решений (проверка простых и многоальтернативных гипотез), оценка параметров и "обучение с учителем", параметрические и непараметрические методы классификации (оценка плотности распределения, Правило ближайших соседей, линейный дискриминант Фишера), нейронные сети, генетические алгоритмы и методы имитационного моделирования. Книга предназначена для специалистов, аспирантов и студентов, изучающих современные методы цифровой обработки сигналов.

Доросинский Леонид Григорьевич, профессор, доктор технических наук, заведующий кафедрой теоретических основ радиотехники Уральского федерального университета имени первого Президента России Б.Н. Ельцина. Автор более 300 работ. Специалист в области обработки сигналов, распознавания образов и изображений.

978-3-659-18960-9

Основы теории принятия решений



Леонид Доросинский

# Основы теории принятия решений

LAP LAMBERT Academic Publishing

Доросинский

**Леонид Доросинский**

**Основы теории принятия решений**

Леонид Доросинский

# Основы теории принятия решений

LAP LAMBERT Academic Publishing

Impressum / **Выходные данные**
Bibliografische Information der Deutschen Nationalbibliothek: Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.d-nb.de abrufbar.


Библиографическая информация, изданная Немецкой Национальной Библиотекой. Немецкая Национальная Библиотека включает данную публикацию в Немецкий Книжный Каталог; с подробными библиографическими данными можно ознакомиться в Интернете по адресу http://dnb.d-nb.de.


Coverbild / Изображение на обложке предоставлено: www.ingimage.com

Verlag / Издатель:
LAP LAMBERT Academic Publishing
ist ein Imprint der / является торговой маркой
OmniScriptum GmbH & Co. KG
Heinrich-Böcking-Str. 6-8, 66121 Saarbrücken, Deutschland / Германия
Email / электронная почта: info@lap-publishing.com

Herstellung: siehe letzte Seite /
Напечатано: см. последнюю страницу
**ISBN: 978-3-659-18960-9**

*Содержание*

# 1. Классическая теория принятия решений

## *1.1. Проверка простых гипотез*

Чтобы пояснить характер задач, с которыми в дальнейшем придется иметь дело, рассмотрим следующий пример.

Допустим, что автомат по приему платежей должен классифицировать денежные купюры нескольких номиналов. Полученное в камере изображение купюры передается на выделитель признаков, назначение которого состоит в уменьшении объема данных при помощи измерения конкретных «признаков» или «свойств», отличающих вид купюр. Далее эти признаки (точнее, измеренные значения этих признаков) подаются на классификатор, предназначенный для оценки представленных данных и принятия окончательного решения относительно номинала купюры.

Другим примером задач может служить цифровая система связи, информация в которой передается путем посылки кодовых комбинаций, содержащих последовательности единиц и нулей. Задача классификатора заключается в принятии – решения определении передаваемой комбинации с минимальными ошибками.

В задачах медицинского диагноза по электрокардиограмме входная информация представлена в виде графика, по виду которого классификатор (в данном случае, как правило, врач) принимает решение о наличии или отсутствии той или иной патологии.

Приведение подобного рода примеров можно продолжать практически бесконечно: это распознавание речи, дактилоскопия, распознавание изображений самой различной природы и т. д.

Начнем рассматривать проблему принятия решений с простейшей задачи классификации изображений двух классов, иначе говоря, с предложения о том, что предъявляемое классификатору изображение может относиться к одному из двух отличающихся классов (фотоснимок суши или водной поверхности, мужчины или женщины, розы или гвоздики), которые мы условно будем

называть первым (гипотеза $H_1$) и нулевым (гипотеза $H_0$). Задача классификатора с минимальной ошибкой принять одно из двух альтернативных решений о наличии изображения первого класса или изображения нулевого класса. Указанное решение должно быть принято в результате анализа принимаемой реализации двумерного случайного поля или (в дискретном варианте) $N$ выборочных значений входного процесса, представляющих собой вектор наблюдаемых данных $\overline{x}=\{x_1,x_2,\ldots x_N\}$. С каждой из вышеупомянутых гипотез связана многомерная плотность вероятности вектора $x$ при условии истинности первой – $p(\overline{x}/H_1)$ и нулевой – $p(\overline{x}/H_0)$ гипотез. Задача синтеза оптимального алгоритма принятия решения состоит в том, чтобы наилучшим образом, с точки зрения выбранного критерия, использовать имеющуюся информацию для принятия решения в пользу той или иной гипотезы.

Прежде чем приступить к ответу на вопрос о том, как распорядиться наблюдаемыми данными, необходимо рассмотреть различные критерии принятия решения.

### *1.2. Критерии минимума среднего риска (критерии Байеса)*

Постановка нашей задачи предполагает, что верна либо гипотеза $H_0$, либо $H_1$. При каждом испытании возможен один из четырех исходов:

1) Верна гипотеза $H_0$, принимаем решение $H_0$.

2) Верна гипотеза $H_0$, принимаем решение $H_1$.

3) Верна гипотеза $H_1$, принимаем решение $H_1$.

4) Верна гипотеза $H_1$, принимаем решение $H_0$.

Первый и третий варианты соответствуют истинным решениям, а второй и четвертый – ошибочным.

Смысл описываемого критерия решения состоит в том, что каждому из возможных исходов приписывается некоторая относительная стоимость (штраф), которая выбирается из эвристических соображений. Обозначим стоимости упомянутых выше исходов наблюдения вектора через $C_{00}$, $C_{10}$, $C_{11}$,

$C_{01}$ соответственно. Первая цифра подстрочного индекса означает выбранную гипотезу (принятое решение), а вторая – гипотезу, которая была правильной.

Следующее допущение состоит в том, что каждой из исходных гипотез соответствуют априорные вероятности $P_1$ и $P_0$.

Каждый опыт, связанный с наблюдением реализации входного случайного процесса, будет сопряжен с определенными потерями, оцененными в форме вышеупомянутых стоимостей. Таким образом, стоимость – случайная величина $C_{ij}\begin{pmatrix} i=0...1 \\ j=0...1 \end{pmatrix}$, вероятность которой есть вероятность события, заключающегося в том, что принимается $i$-я гипотеза, а справедлива $j$-я. Необходимо сформулировать критерии принятия решения таким образом, чтобы минимизировать среднюю величину стоимости, иначе говоря, минимизировать средний риск:

$$R = \sum_{i=0}^{1}\sum_{j=0}^{1} C_{ij} P_{ij}, \qquad (1.1)$$

где $P_{ij}$ – вероятность совместного выполнения двух событий: принимается решение в пользу $i$ - й гипотезы, а истинной является $j$ - я.

Использование формулы умножения вероятностей позволяет получить выражение

$$P_{ij} = P_j P(H_i / H_j), \qquad (1.2)$$

где $P(H_i / H_j)$ – условная плотность вероятности принять решение в пользу $i$-й гипотезы при истинной $j$-й.

С учетом (1.2) запишем выражение среднего риска:

$$R = C_{00}P_0P(H_0/H_0) + C_{10}P_0P(H_1/H_0) + C_{01}P_1P(H_0/H_1) + C_{11}P_1P(H_1/H_1). \qquad (1.3)$$

Поскольку мы предполагаем, что в итоге следует выбрать либо $H_1$, либо $H_0$, правило решения заключается в разбиении пространства наблюдения $\Gamma(\bar{x} \in \Gamma)$ на две части: $\Gamma_1$ и $\Gamma_0$ (рис. 1.1). Если результат наблюдения вектор $\bar{x}$ оказывается в $\Gamma_0$, то принимается решение в пользу гипотезы $H_0$, а если в $\Gamma_1$ – то в пользу $H_1$.

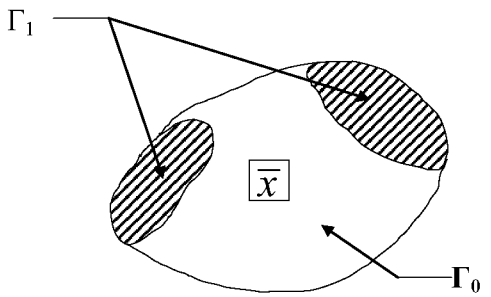



Рис. 1.1

Теперь запишем выражение для риска через условные вероятности и подпространства решения:

$$R = C_{00}P_0\int_{\Gamma_0} p(\overline{x}/H_0)dx + C_{10}P_0\int_{\Gamma_1} p(\overline{x}/H_0)dx + C_{11}P_1\int_{\Gamma_1} p(\overline{x}/H_1)dx + C_{01}P_1\int_{\Gamma_0} p(\overline{x}/H_1)d\overline{x}\,. \quad (1.4)$$

При записи формулы (1.4) учтено, что условная вероятность принятия *i*-й гипотезы при истинной *j*-й совпадает с вероятностью попадания вектора $\overline{x}$, распределенного с плотностью $p(\overline{x}/H_j)$, в подпространство $\Gamma_i$ пространства $\Gamma$ ($\Gamma_i \subset \Gamma$).

Далее будем считать, что стоимость ошибочно принятого решения выше, чем стоимость правильного решения, т. е.

$$C_{10} > C_{00};\ C_{01} > C_{11}. \quad (1.5)$$

Чтобы определить результат байесовского испытания (минимизировать средний риск), следует выбрать подпространства решений $\Gamma_0$ и $\Gamma_1$ так, чтобы величина (1.4) была сведена к минимуму.

Поскольку все пространство $\Gamma$ есть сумма $\Gamma_0$ и $\Gamma_1$

$$\Gamma = \Gamma_0 + \Gamma_1, \quad (1.6)$$

постольку перепишем выражение риска в следующем виде:

$$R = C_{00}P_0\int_{\Gamma_0} p(\overline{x}/H_0)d\overline{x} + C_{10}P_0\int_{\Gamma-\Gamma_0} p(\overline{x}/H_0)d\overline{x} + C_{11}P_1\int_{\Gamma-\Gamma_0} p(\overline{x}/H_1)d\overline{x} + C_{01}P_1\int_{\Gamma_0} p(\overline{x}/H_1)d\overline{x} \quad (1.7)$$

Учитывая условия нормировки

$$\int_{\Gamma} p(\bar{x}/H_1)d\bar{x} = \int_{\Gamma} p(\bar{x}/H_0)d\bar{x} = 1, \qquad (1.8)$$

выражение (1.7) можно привести к виду

$$R = C_{10}P_0 + C_{11}P_1 + \int_{\Gamma_0} [P_1(C_{01} - C_{11})p(\bar{x}/H_1) - P_0(C_{10} - C_{00})p(\bar{x}/H_0)]d\bar{x} \qquad (1.9)$$

Первые два члена в (1.9) не изменяются, коэффициенты $P_1(C_{01} - C_{11})$ и $P_0(C_{10} - C_{00})$ в силу предположения (1.5) положительны. Поэтому для минимизации среднего риска область $\Gamma_0$ принятия нулевой гипотезы должна быть выбрана таким образом, что все значения $\bar{x}$, при которых второй член подынтегрального выражения больше, чем первый, были включены в эту область, т.к. эти значения вектора наблюдаемых данных вносят отрицательный вклад в интеграл и, следовательно, уменьшают средний риск.

Аналогично все значения $\bar{x}$, когда второй член подынтегрального выражения меньше первого, следует исключить из $\Gamma_0$ (отнести к $\Gamma_1$), поскольку ими вносится в интеграл положительная величина. Таким образом, области решения определяются следующим условием:

если

$$P_1(C_{01} - C_{11})p(\bar{x}/H_1) > P_0(C_{10} - C_{00})p(\bar{x}/H_0), \qquad (1.10)$$

то относим $\bar{x}$ к $\Gamma_1$ и, следовательно, утверждаем, что истинна гипотеза $H_1$; в противном случае относим $\bar{x}$ к $\Gamma_0$ и утверждаем справедливость $H_0$. Перепишем формулу (1.10) в виде

$$\frac{p(\bar{x}/H_1)}{p(\bar{x}/H_0)} \triangleright\triangleleft \frac{P_0(C_{10}\ C_{00})}{P_1(C_{01}\ C_{11})}. \qquad (1.11)$$

Величину в левой части неравенства (1.11) называют отношением правдоподобия и обозначают через $\Lambda(\bar{x})$:

$$\Lambda(\bar{x}) = \frac{p(\bar{x}/H_1)}{p(\bar{x}/H_0)}. \qquad (1.12)$$

Величина в правой части неравенства (1.11) является порогом испытания и обозначается через $\eta$:

$$\eta = \frac{P_0(C_{10} - C_{00})}{P_1(C_{01} - C_{11})}. \qquad (1.13)$$

Таким образом, критерий Байеса приводит нас к критерию отношения правдоподобия

$$\Delta(\bar{x}) \lessgtr \eta \qquad (1.14)$$

Важно отметить, что необходимые для минимизации среднего риска функциональные преобразования над наблюдаемыми данными заключаются в вычислении $\Lambda(\bar{x})$, а значения априорных вероятностей и стоимостей учитываются только при определении порога. Указанная инвариантность процедуры обработки информации имеет большое практическое значение. Условие (1.14) позволяет определить все устройство обработки, рассматривая $\eta$ как переменный порог, учитывающий происходящие изменения в наших оценках априорных вероятностей и стоимостей.

Если $\varphi[\Lambda(\bar{x})]$ – монотонная функция, то эквивалентной формой записи критерия отношения правдоподобия будет

$$\phi[\Delta(\bar{x})] \lessgtr \phi(\eta).$$

В том случае, когда отношение правдоподобия принадлежит к экспоненциальному семейству функций, в качестве функции $\varphi$ целесообразно выбрать натуральный логарифм:

$$\ln \Delta(\bar{x}) \lessgtr \ln(\eta).$$

При этом устройство классификации двух изображений существенно упрощается.

### *1.3. Многоальтернативная проверка гипотез*

Рассмотрим процедуру принятия решения при наличии нескольких вариантов: изображения нескольких классов, кодовые комбинации, соответствующие нескольким передаваемым сигналам, дактилоскопические исследования группы людей, распознавание букв русского алфавита и т. п.

Итак, предположим, что имеется $M^2$ альтернатив, причем каждой из них априори приписывают некоторую стоимость и полагают, что задана система априорных вероятностей $P_0, P_1, \ldots, P_{M-1}$ для каждой из M возможных гипотез.

Для отыскания байесовского правила решения обозначим названную выше стоимость каждого образа действий через $C_{ij}$ ($i$-я гипотеза выбрана в качестве истинной, а $j$-я гипотеза является истинной на самом деле).

Область пространства, в которой мы выбираем гипотезу $H_i$, обозначим $\Gamma_i$.

Запишем выражение для риска:

$$R = \sum_{i=0}^{M-1} \sum_{j=0}^{M-1} P_j C_{ij} \int_{\Gamma_m} p(\overline{x} / H_j) d\overline{x} .$$

Решение будет оптимальным при таком разбиении пространства Г на М непересекающихся подпространств, чтобы риск был минимален.

Метод решения данной задачи не отличается от предыдущего. Проиллюстрируем обобщение для М = 3.

Поскольку области не пересекаются, постольку $\Gamma_0 = \Gamma - \Gamma_1 - \Gamma_2$. Запишем выражение для риска.

$$\begin{aligned}
R = {} & P_0 C_{00} \int_{\Gamma - \Gamma_1 - \Gamma_2} p(\overline{x} / H_0) d\overline{x} + P_0 C_{10} \int_{\Gamma_{12}} p(\overline{x} / H_0) d\overline{x} + P_0 C_{20} \int_{\Gamma_2} p(\overline{x} / H_0) d\overline{x} + \\
& + P_1 C_{01} \int_{\Gamma_0} p(\overline{x} / H_1) d\overline{x} + P_1 C_{11} \int_{\Gamma - \Gamma_0 - \Gamma_2} p(\overline{x} / H_1) d\overline{x} + P_1 C_{21} \int_{\Gamma_2} p(\overline{x} / H_1) d\overline{x} + \\
& + P_2 C_{02} \int_{\Gamma_0} p(\overline{x} / H_2) d\overline{x} + + P_2 C_{12} \int_{\Gamma_1} p(\overline{x} / H_2) d\overline{x} + P_2 C_{22} \int_{\Gamma - \Gamma 0_1 - \Gamma_2} p(\overline{x} / H_2) d\overline{x} = \\
= {} & P_0 C_{00} + P_1 C_{11} + P_2 C_{22} + \int_{\Gamma_0} [P_1 C_{01} p(\overline{x} / H_1) - P_1 C_{11} p(\overline{x} / H_1) + P_2 C_{02} p(\overline{x} / H_2) - \\
& - P_2 C_{22} p(\overline{x} / H_2)] d\overline{x} + \int_{\Gamma_1} [P_0 C_{10} p(\overline{x} / H_0) - P_0 C_{00} p(\overline{x} / H_0) + P_2 C_{12} p(\overline{x} / H_2) - \\
& - P_2 C_{22} p(\overline{x} / H_2)] d\overline{x} + \int_{\Gamma_2} [P_0 C_{20} p(\overline{x} / H_0) - P_0 C_{00} p(\overline{x} / H_0) + P_1 C_{21} p(\overline{x} / H_1) - P_1 C_{11} p(\overline{x} / H_1)] d\overline{x} \quad .
\end{aligned} \tag{1.15}$$

После несложных преобразований получим

$$\begin{aligned}
R = {} & P_0 C_{00} + P_1 C_{11} + P_2 C_{22} + \int_{\Gamma_0} [P_1 (C_{01} - C_{11}) p(\overline{x} / H_1) + P_2 (C_{02} - C_{22}) p(\overline{x} / H_2)] d\overline{x} + \\
& + \int_{\Gamma_1} [P_0 (C_{10} - C_{00}) p(\overline{x} / H_0) + P_2 (C_{12} - C_{22}) p(\overline{x} / H_2)] d\overline{x} + \\
& + \int_{\Gamma_2} [P_0 (C_{20} - C_{00}) p(\overline{x} / H_0) + P_1 (C_{21} - C_{11}) p(\overline{x} / H_1)] d\overline{x}.
\end{aligned}$$

Решение принимается в пользу той гипотезы, для которой соответствующее подынтегральное выражение в формуле (1.15) меньше:

$$P_1(C_{01}-C_{11})p(\overline{x}/H_1)+P_2(C_{02}-C_{22})p(\overline{x}/H_2)\overset{H_0\ H_1}{\lessgtr} P_0(C_{10}-C_{00})p(\overline{x}/H_0)+P_2(C_{12}-C_{22})p(\overline{x}/H_2);$$

$$P_1(C_{01}-C_{11})p(\overline{x}/H_1)+P_2(C_{02}-C_{22})p(\overline{x}/H_2)\overset{H_0\ H_2}{\lessgtr} P_0(C_{20}-C_{00})p(\overline{x}/H_0)+P_1(C_{21}-C_{11})p(\overline{x}/H_1);$$

$$P_0(C_{10}-C_{00})p(\overline{x}/H_0)+P_2(C_{12}-C_{22})p(\overline{x}/H_2)\overset{H_1\ H_2}{\lessgtr} P_0(C_{20}-C_{00})p(\overline{x}/H_0)+P_1(C_{21}-C_{11})p(\overline{x}/H_1). \quad (1.16)$$

Определим отношение:

$$\Delta_i(\overline{x})=\frac{p_i(\overline{x})}{p_0(\overline{x})}. \quad (1.17)$$

и перепишем правило решения (1.20) в виде:

$$P_1(C_{01}-C_{11})\Delta_1(\overline{x})\overset{H_0\ H_1}{\lessgtr} P_0(C_{10}-C_{00})+P_2(C_{12}-C_{22})\Delta_2(\overline{x});$$

$$P_2(C_{02}-C_{22})\Delta_2(\overline{x})\overset{H_0\ H_2}{\lessgtr} P_0(C_{20}-C_{00})+P_1(C_{21}-C_{11})\Delta_1(\overline{x});$$

$$P_2(C_{12}-C_{22})\Delta_2(\overline{x})\overset{H_1\ H_2}{\lessgtr} P_0(C_{20}-C_{00})+P_1(C_{21}-C_{11})\Delta_1(\overline{x}).$$

Рассмотрим частный случай, когда стоимости ошибочных решений одинаковы и равны единице, а стоимости верных решений считаются равными нулю:

$$C_{ij}=1-\delta_{ij}, \quad (1.18)$$

где $\delta_{ij}$ - символ Кронекера, $\delta_{ij}=\begin{cases}1 \text{ при } i=j;\\ 0 \text{ при } i\neq j.\end{cases}$

При этом получаем простое правило решения:

$$P_1\Delta_1(\overline{x})\overset{H_0\ H_1}{\lessgtr} P_0;$$

$$P_2\Delta_2(\overline{x})\overset{H_0\ H_2}{\lessgtr} P_0;$$

$$P_2\Delta_2(\overline{x})\overset{H_1\ H_2}{\lessgtr} P_1\Delta_1(\overline{x}).$$

Подставляя (1.17) в (1.18), приходим к следующему правилу решения:

$$P_1p(\overline{x}/H_1)\overset{H_0\ H_1}{\lessgtr} P_0p(\overline{x}/H_0);$$

$$P_2p(\overline{x}/H_2)\overset{H_0\ H_2}{\lessgtr} P_0p(\overline{x}/H_0);$$

$$P_2p(\overline{x}/H_2)\overset{H_1\ H_2}{\lessgtr} P_1p(\overline{x}/H_1).$$

В этом случае оптимальный критерий классификации заключается в формировании апостериорной плотности вероятности для каждого возможного класса и выбора максимума.

Обобщение на случай произвольного числа классов M очевидно.

Решение о выборе номера класса производится путем отыскания максимума апостериорной плотности:

$$\hat{i} = \max_i \left[ P_i \, p(\bar{x} / H_i) \right] = \max_i g_i(\bar{x}) .$$

## 2. Классификаторы, разделяющие функции и поверхности решений

### *2.1. Случай многих классов*

Канонической формой классификатора может служить его представление в виде системы разделяющих функций $g_i(\bar{x})$. Классификатор ставит вектор признаков $\bar{x}$ в соответствие гипотезе (классу) $H_i$, если для всех $j \neq i$ справедливо неравенство

$$g_i(\bar{x}) \rhd g_j(\bar{x}) .$$

Классификатор, таким образом, рассматривается как устройство, вычисляющее M разделяющих функций и выбирающее решение, соответствующее наибольшей из них.

Очевидно, что выбор разделяющих функций не единственен. Всегда можно, не влияя на решение, умножить разделяющие функции на положительную константу или прибавить к ним какую-либо константу. Более того, если заменить каждую из $g_i(\bar{x})$ на $f(g_i(\bar{x}))$, где $f(\bullet)$ — монотонно возрастающая функция, то результат классификации не изменится. Это обстоятельство может привести к существенным аналитическим и расчетным упрощениям. В частности, при классификации с минимальным уровнем наиболее удобным на практике вариантом разделяющей функции является

$$g_i(\bar{x}) = \log p(\bar{x}/H_i) + \log P(H_i). \qquad (2.1)$$

Решающие правила остаются эквивалентными. Действие решающего правила заключается в разбиении пространства признаков $\bar{x}$ на M областей решений $\Gamma_1, \Gamma_2, \ldots, \Gamma_M$. Уравнение границы, разделяющей области $\Gamma_i$ и $\Gamma_j$ имеет вид

$$g_i(\bar{x}) = g_j(\bar{x}).$$

## 2.2. Вероятности ошибок и интегралы ошибок

Рассмотрим случай двух классов. Возможно два типа ошибок классификации:

- вектор наблюдаемых данных попадает в пространство $\Gamma_1$ в то время, как истинное состояние природы соответствует гипотезе $H_0$;
- вектор наблюдаемых данных попадает в пространство $\Gamma_0$, хотя истинное состояние природы – $H_1$.

В связи с тем, что названные события взаимоисключающие и составляют полное множество событий, вероятность ошибки может быть рассчитана в соответствии с выражением

$$\begin{aligned} P_{ошибки} &= P(\bar{x} \in \Gamma_1, H_0) + P(\bar{x} \in \Gamma_0, H_1) = \\ &= P(\bar{x} \in \Gamma_1 / H_0) P(H_0) + P(\bar{x} \in \Gamma_0, H_1) P(H1) = \\ &= P(H_0) \int_{\Gamma_2} p(\bar{x} / H_0)\, d\bar{x} + P(H_1) \int_{\Gamma_0} p(\bar{x} / H_1)\, d\bar{x} \ . \end{aligned}$$

В случае многих классов проще вычислить вероятность правильного решения:

$$\begin{aligned} P_{правильного\ решения} &= \sum_{i=1}^{M} P(\bar{x} \in \Gamma_i, H_i) = \sum_{i=1}^{M} P(\bar{x} \in \Gamma_i / H_i) P(H_i) = \\ &= P(H_i) \sum_{i=1}^{M} \int_{\Gamma_i} p(\bar{x} \in \Gamma_i)\, d\bar{x}. \end{aligned}$$

Полученный результат справедлив при любом разбиении пространства решений. Разбиение в соответствии с байесовским критерием гарантирует, что полученная вероятность будет максимальной.

## 2.3. Правило принятия решения при нормальной плотности вероятностей признаков

В соответствии с выражением (2.1) структура байесовского алгоритма принятия решения определяется в основном видом условных плотностей $p(\bar{x}/H_i)$. Из множества различных функций плотности наибольшее значение имеет многомерная нормальная плотность распределения по следующим соображениям.

1. Многочисленные эксперименты и практические исследования говорят о чрезвычайно широком распространении названной плотности во многих практических ситуациях.

2. В силу центральной предельной теоремы при условии, что рассматриваемый признак есть по существу результат сложения большого количества примерно равнозначных явлений (ток – сумма электронов, тепловое движение – сумма перемещений атомов и т. п.), его распределение, по крайней мере в асимптотике, стремится к нормальному.

3. Многомерная нормальная плотность распределения дает подходящую модель для одного важного случая, когда значения векторов признаков $x$ для данного класса представляются непрерывнозначными, слегка искаженными версиями единственного типичного вектора, или вектора-прототипа, $\mu$. Именно этого ожидают, когда классификатор выбирается так, чтобы выделять те признаки, которые, будучи различными для образов, принадлежащих различным классам, были бы, возможно, более схожи для образов из одного и того же класса.

4. Немаловажную роль при использовании нормальной плотности играет удобство ее аналитического представления и операций над ней.

Многомерная нормальная плотность распределения в общем виде представляется выражением

$$p(y)=\frac{1}{(2\pi)^{d/2}|R|^{1/2}}\exp\left[-\frac{1}{2}(\bar{x}-\bar{\mu})^t R^{-1}(\bar{x}-\bar{\mu})\right], \qquad (2.2)$$

где $\bar{x}$ есть d-компонентный вектор-столбец, $\bar{\mu}$ – есть d-компонентный вектор среднего значения, $R$ – ковариационная матрица размера d×d, $(\bar{x}-\bar{\mu})^t$ – транспонированный вектор $(\bar{x}-\bar{\mu})$, $R^{-1}$ – матрица, обратная $R$, а $|R|$ – детерминант матрицы $R$. Для простоты выражение (2.2) часто записывается сокращенно в виде $p(x)\sim N(\bar{\mu}, R)$.

Вектор $\bar{\mu} = M[\bar{x}]$ представляет собой вектор математических ожиданий вектора $\bar{x}$, а матрица $R = M[(\bar{x}-\bar{\mu})(\bar{x}-\bar{\mu})^t)]$ – матрицу ковариаций вектора $\bar{x}$ ($M[\ ]$ – операция вычисления математического ожидания).

Ковариационная матрица $R$ всегда симметрична и положительно полуопределена. Ограничимся рассмотрением случаев, когда $R$ положительно определена, так что ее детерминант строго положителен. Диагональный элемент матрицы $R$ представляет собой дисперсию $R_{ii} = \sigma_{ii}$, а недиагональный элемент $R_{ij}$ есть ковариация $x_i$ и $x_j$. Если $x_i$ и $x_j$ статистически независимы, то $R_{ij} = 0$. Если все недиагональные элементы равны нулю, то $p(\bar{x})$ сводится к произведению одномерных нормальных плотностей компонент вектора $\bar{x}$.

Многомерная нормальная плотность распределения полностью определяется $d+d(d+1)/2$ параметрами — элементами вектора среднего значения $\bar{\mu}$ и независимыми элементами ковариационной матрицы $R$. Выборки нормально распределенной случайной величины имеют тенденцию попадать в одну область или кластер. Центр кластера определяется вектором среднего значения, а форма — ковариационной матрицей. Из соотношения (2.2) следует, что точки постоянной плотности образуют гиперэллипсоиды, для которых квадратичная форма $(\bar{x}-\bar{\mu})^t R^{-1}(\bar{x}-\bar{\mu})$ постоянна. Главные оси этих гиперэллипсоидов задаются собственными векторами $R$, причем длины осей определяются собственными значениями. Величину

$$r^2 = (\bar{x}-\bar{\mu})^t R^{-1} (\bar{x}-\bar{\mu})$$

называют квадратичным махаланобисовым расстоянием от $\bar{x}$ до $\bar{\mu}$. Линии постоянной плотности, таким образом, представляют собой гиперэллипсоиды постоянного махаланобисова расстояния $\bar{\mu}$. Объем этих гиперэллипсоидов служит мерой разброса выборок относительно среднего значения.

Как мы показали выше, классификация с минимальным уровнем ошибки может осуществляться посредством разделяющих функций вида

$$g_i(\bar{x}) = \log p(\bar{x}/H_i) + \log P(H_i).$$

Когда многомерная плотность $p(\bar{x}/H_i)$ нормальна, согласно выражению (2.2) получаем

$$g_i(\bar{x}) = -\frac{1}{2}(\bar{x}-\bar{\mu}_i)^t R_i^{-1}(\bar{x}-\bar{\mu}_i) - \frac{1}{2}\log|R_i| + \log P(H_i). \qquad (2.3)$$

Последнее слагаемое определяется априорными вероятностями гипотез. Мы его учитывать не будем, т.к. названное слагаемое может быть добавлено на любом этапе работы алгоритма.

Рассмотрим ряд частных случаев.

1. Признаки статистически независимы и имеют одинаковую дисперсию $R = \sigma^2 E$, где $E$ – единичная матрица.

В этом случае разделяющая функция сводится к вычислению эвклидова расстояния между вектором признаков и каждым вектором математических решений. Решение выбирается в пользу той гипотезы, для которой названное расстояние минимально.

$$\hat{i} = \frac{1}{2\sigma^2} \min \left\| \bar{x} - \bar{\mu}_i \right\|^2, \tag{2.4}$$

где

$$\left\| \bar{x} - \bar{\mu}_i \right\|^2 = (\bar{x} - \bar{\mu}_i)^t (\bar{x} - \bar{\mu}_i) = \sum_{k=1}^{d} (x_k - \mu_i^k)^2. \tag{2.5}$$

Такой классификатор называют классификатором по минимуму расстояния. Если каждый из векторов средних значений считать идеальным прототипом или эталоном для образов своего класса, то это по существу будет процедура сравнения с эталоном.

Если априорные вероятности не равны, то, согласно соотношениям (2.3) и (2.4), квадрат расстояния (2.5) должен быть нормирован по дисперсии (поделен на $2\sigma^2$) и смещен на величину $log\ P(H_i)$; поэтому в случае, когда вектор $\bar{x}$ одинаково близок к двум различным векторам средних значений, при принятии решения следует предпочесть класс, априори более вероятный.

Произведя перемножение в формуле (2.3) и отбросив одинаковое для всех i слагаемое, приходим к линейной разделяющей функции:

$$g_i(\bar{x}) = \bar{w}_i^t \bar{x} + w_{i0}, \tag{2.6}$$

где

$$\bar{w}_i = \frac{1}{\sigma^2} \bar{\mu}_i;$$

$$w_{i0} = -\frac{1}{2\sigma^2} \bar{\mu}_i^t \bar{\mu}_i + \log P(H_i).$$

Классификатор, основанный на использовании линейных разделяющих функций, называется линейной машиной.

2. Ковариационные матрицы для всех классов одинаковы. $R_i = R$.

Это соответствует ситуации, при которой выборки попадают внутрь гиперэллипсоидальных областей (кластеров) одинаковых размеров и формы, с вектором средних значений в центре каждой.

После того как мы пренебрегаем не зависящими от $i$ слагаемыми, получаем разделяющие функции вида

$$g_i(\overline{x}) = -\frac{1}{2}(\overline{x} - \overline{\mu}_i)^t R^{-1}(\overline{x} - \overline{\mu}_i) + \log P(H_i). \qquad (2.7)$$

Если априорные вероятности для всех M классов равны, то последним слагаемым в формуле (2.7) можно пренебречь. Оптимальное решающее правило в таком случае снова оказывается очень простым: для классификации вектора признаков следует определить квадратичное махаланобисово расстояние от $\overline{x}$ до каждого из M векторов средних значений и отнести $\overline{x}$ к классу, соответствующему ближайшему среднему значению. Как и прежде, в случае неравных априорных вероятностей, при принятии решения несколько большее предпочтение отдается классу, априори более вероятному.

После раскрытия квадратичной формы и отбрасывания слагаемых, не изменяющихся при разных значениях $i$, получаем выражения:

$$g_i(\overline{x}) = \overline{w}_i^t \overline{x} + w_{i0}; \qquad (2.8)$$

$$\overline{w}_i = R^{-1}\overline{\mu}_i;$$

$$w_{i0} = -\frac{1}{2}\overline{\mu}_i^t R^{-1}\overline{\mu}_i + \log P(H_i).$$

3. Произвольные корреляционные матрицы $R_i$.

В общем случае многомерного нормального распределения ковариационные матрицы для каждого класса разные. В этом случае разделяющие функции получаются квадратичными:

$$g_i(\overline{x}) = \overline{x}^t W_i \overline{x} + \overline{w}_i^t \overline{x} + w_{i0}, \qquad (2.9)$$

где

$$W_i = -\frac{1}{2}R_i^{-1};$$

$$\overline{w}_i = R_i^{-1}\overline{\mu}_i;$$

$$w_{i0} = -\frac{1}{2}\bar{\mu}_i^t R_i^{-1}\bar{\mu}_i - \frac{1}{2}\log|R_i| + \log P(H_i).$$

Таким образом, в зависимости от ситуации (независимые признаки, одинаковые ковариационные матрицы, различающиеся ковариационные матрицы) решение принимается в пользу той гипотезы, для которой выражение решающей функции (2.6), (2.8) и (2.9) соответственно максимально.

# 3. Оценка параметров и обучение с учителем

## *3.1. Оценка по максимуму правдоподобия*

В практических условиях апостериорные плотности вероятностей $P(H_i/\bar{x})$, как правило, либо неизвестны вообще, либо известны с точностью до ряда параметров. В то же время обычно имеется набор так называемых обучающих выборок, достоверно принадлежащих каждому из распознаваемых классов. Число этих выборок зачастую достаточно мало, чтобы вынести решение о функциональном виде требуемых плотностей вероятностей, но достаточно велико для построения оценок параметров названных плотностей, если их функциональный вид предполагается известным.

Допустим, что есть основания предполагать, что плотность вероятности $p(\bar{x}/H_i)$ имеет нормальное распределение со средним значением $\bar{\mu}_i$ и ковариационной матрицей $R_i$, хотя точные значения названных величин точно неизвестны.

В этом случае решение принимается в соответствии с теми же принципами и правилами, что и в главе 2, где в формулы (2.6), (2.8) и (2.9) подставляются не точно известные значения $\bar{\mu}_i$ и $R_i$, а их оптимальные в каком-то смысле оценки. Среди таких оценок наилучшими в практических ситуациях свойствами обладают оценки, полученные по методу максимального правдоподобия.

Рассмотрим оценку по методу максимального правдоподобия. Предположим, что множество имеющихся обучающих выборок разбито на M

классов $X_1, X_2, \ldots, X_M$, причем выборки в каждом $X_i$ статистически независимы и имеют плотность распределения $p(\bar{x}/H_i)$. Будем считать, что плотность $p(\bar{x}/H_i)$ задана в параметрической форме, т. е. известно ее аналитическое выражение с точностью до неизвестного векторного параметра $\Theta$. Например, нам известно, что выборки подчиняются нормальному закону распределения с неизвестным вектором математических ожиданий $\bar{\mu}_i$ и ковариационной матрицей $R_i$. В этом случае компоненты вектора $\Theta$ составлены из компонент $\bar{\mu}_i$ и $R_i$.

Для того чтобы в явном виде показать зависимость от неизвестных параметров, запишем плотность вероятностей в виде $p(\bar{x}/H_i, \Theta_i)$. Задача оценки неизвестных параметров заключается в определении их величин по наблюдаемым данным наилучшим образом.

Будем считать, что выборки, принадлежащие наблюдаемым данным $X_i$, не содержат информации о векторе параметров $\Theta_j$, т. е. предполагается функциональная независимость параметров, принадлежащих разным классам. Последнее обстоятельство дает возможность рассматривать отдельно каждый класс.

Пусть $X$ содержит $n$ выборок $X = \{\bar{x}_1, \bar{x}_2 \ldots\ldots \bar{x}_n\}$ Так как выборки получены независимо, имеем:

$$p(\bar{x}/\Theta) = \prod_{k=1}^{n} p(x_k/\Theta). \qquad (3.1)$$

Рассматриваемая как функция от $\Theta$ плотность $p(\bar{x}/\Theta)$ называется функцией правдоподобия. Оценка по максимуму правдоподобия величины $\Theta$ есть такая величина $\hat{\Theta}$, при которой выражение (3.1) максимально.

На практике эквивалентным, но более простым по своим вычислениям является отыскание максимума не собственно плотности вероятности, а ее логарифма. Если ввести оператор градиента:

$$\Delta_{\Theta}=\begin{bmatrix}\dfrac{\partial}{\partial\Theta_1}\\ \cdot\\ \cdot\\ \cdot\\ \dfrac{\partial}{\partial\Theta_p}\end{bmatrix},$$

где $p$ – размерность вектора параметров и обозначить функцию логарифма правдоподобия:

$$l(\Theta) = \log p(\overline{x}/\Theta),$$

то оптимальная оценка вектора параметров $\Theta$ может быть получена из решения уравнения:

$$\Delta_{\Theta}(l) = \sum_{k=1}^{n} \Delta_{\Theta} \log p(x_k / \Theta).$$

Применим полученные результаты для многомерного нормального распределения. Начнем со случая, когда неизвестно только среднее значение. Запишем выражение для логарифма функции правдоподобия и оператора градиента:

$$\log p(x_k/\mu) = -\frac{1}{2}\log\left\{(2\pi)^d |R|\right\} - \frac{1}{2}(x_k-\mu)^t R^{-1}(x_k-\mu);$$

$$\Delta_{\mu} \log p(x_k/\mu) = R^{-1}(x_k-\mu).$$

С учетом (3.1) получаем уравнение

$$\sum_{k=1}^{n} R^{-1}(x_k - \hat{\mu}) = 0,$$

из которого следует:

$$\hat{\mu} = \frac{1}{n}\sum_{k=1}^{n} x_k. \qquad (3.2)$$

Полученный результат свидетельствует о том, что оценка по максимуму правдоподобия равна среднему арифметическому выборок. Для ковариационной функции аналогичным образом может быть получена ее оценка по максимуму правдоподобия, которая имеет вид:

$$\hat{R} = \frac{1}{n}\sum_{k=1}^{n}(x_k-\hat{\mu})(x_k-\hat{\mu})^t. \qquad (3.3)$$

Полученные результаты (3.2) и (3.3) представляются совершенно естественными и интуитивно понятными.

В результате процедура классификации выглядит следующим образом.

1. Этап обучения. По обучающим выборкам строятся оценки математических ожиданий и ковариационных функций (формулы (3.2) и (3.3)) для каждой из M конкурирующих гипотез.

2. Далее вычисляются M решающих функций (формулы (2.6), (2.8), (2.9).

3. Решение принимается в пользу той гипотезы, для которой решающая функция максимальна.

Рассматриваемые решающие функции, основанные на вычислении взвешенных расстояний между наблюдаемым вектором признаков и вектором математических ожиданий, полученным на этапе обучения, дают хорошие результаты в случае, когда вектор признаков имеет нормальное или близкое к нему распределение.

### *3.2. Байесовский классификатор*

Из курса математической статистики известно, что оценка по максимуму правдоподобия для ковариационной матрицы смещена, т. е. ожидаемое значение $\hat{R}$ не равно $R$. Несмещенная оценка для $R$ задается выборочной ковариационной матрицей

$$\hat{R}=\frac{1}{n-1}\sum_{k=1}^{n}(x_k - \ )(x_k - \ )^t. \qquad (3.4)$$

Очевидно, что оценки (3.3) и (3.4), совпадают при большом $n$.

Наличие двух сходных и тем не менее разных оценок для ковариационной матрицы смущает многих исследователей, т.к. возникает вопрос: какая же из них “верная”? Ответить на это можно, сказав, что каждая из этих оценок ни верна, ни ложна: они просто различны. Наличие двух различных оценок показывает, что единой оценки, включающей все свойства, которые только можно пожелать, не существует. Для наших целей сформулировать наиболее желательные свойства довольно сложно — нам

нужна такая оценка, которая позволила бы наилучшим образом проводить классификацию. Хотя разрабатывать классификатор, используя оценки по максимуму правдоподобия для неизвестных параметров, обычно представляется разумным и логичным, вполне естествен вопрос, а нет ли других оценок, обеспечивающих еще лучшее качество работы. В данном разделе мы рассмотрим этот вопрос с байесовской точки зрения.

Сущность байесовской классификации заложена в расчете апостериорной вероятности $P(H_i/\overline{x})$. Байесовское правило позволяет вычислить эти вероятности по априорным вероятностям $P(H_i)$ и условным по классу вероятностям $p(\overline{x}/H_i)$. Однако возникает вопрос: как быть, если названные вероятности неизвестны. Для ответа на него мы должны вычислить $P(H_i/\overline{x})$, максимально используя всю информацию, которая есть в нашем распоряжении. Часть такой информации может быть априорной, часть содержаться в множестве выборок. Пусть $X$ означает множество выборок. Применив формулу Байеса, получим апостериорную плотность вероятности

$$P(H_i/\overline{x},X) = \frac{\mathrm{p}(\overline{x}/\mathrm{H_i},X)P(H_i)}{\sum_{j=1}^{M}\mathrm{p}(\overline{x}/\mathrm{H_j},X)P(H_j)}. \qquad (3.5)$$

В данном выражения (3.10) мы считаем, что значения априорных вероятностей известны и не зависят от выборок $X$. Кроме того, предполагаем, что выборки можно разделить по классам на M подмножеств: $X_1, X_2, \ldots\ldots X_M$, причем выборки $X_i$ принадлежат $H_i$. и не оказывают влияния на все $p(\overline{x}/H_i, X)$, если $i \neq j$. Принятые предположения позволяют записать уравнение (3.10) в виде

$$P(H_i/\overline{x},\mathrm{X}) = \frac{\mathrm{p}(\overline{x}/\mathrm{H_i},X_i)P(H_i)}{\sum_{j=1}^{M}\mathrm{p}(\overline{x}/\mathrm{H_j},X_j)P(H_j)}. \qquad (3.6)$$

Для принятия решения по правилу (3.6), например, путем выбора максимума, мы должны решить M задач оценки M плотностей вероятностей $p(\overline{x}/X_i)$ по $M$ обучающим выборкам.

В качестве примера решим задачу оценки неизвестного вектора средних значений $\bar{\mu}$. Начнем анализ с одномерного случая, когда $p(x/\mu)$ представляет собой нормальное распределение с математическим ожиданием, равным $\mu$, и дисперсией $\sigma^2$ и общее число выборок равно $n$: $X = \{x_1, x_2, \ldots\ldots x_n\}$.

Предположим, что априорные сведения о среднем значении $\mu$ исчерпываются известной априорной плотностью, которая также считается нормальной с математическим ожиданием $\mu_0$ и дисперсией $\sigma_0^{2}$.

Для определения среднего значения воспользуемся байесовским правилом. Апостериорная плотность параметра имеет вид

$$p(\mu / X) = \frac{p(X/\mu)p(\mu)}{\int p(X/\mu)p(\mu)d\mu} = \alpha \prod_{\kappa=1}^{n} p(X_k/\mu)p(\mu), \qquad (3.7)$$

где $\alpha$ масштабный множитель, не зависящий от $\mu$.

Подставляя в выражение (3.7) соответствующие нормальные плотности вероятностей, получаем

$$p(\mu / X) = \alpha \prod_{\kappa=1}^{n} \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{1}{2}\left(\frac{x_k-\mu}{\sigma}\right)^2\right] * \frac{1}{\sqrt{2\pi}\sigma_0} \exp\left[-\frac{1}{2}\left(\frac{\mu-\mu_0}{\sigma_0}\right)^2\right] =$$

$$= \alpha' \exp\left[-\frac{1}{2}\left\{\sum_{k=1}^{n}\left(\frac{\mu-x_k}{\sigma}\right)^2 + \left(\frac{\mu-\mu_0}{\sigma_0}\right)^2\right\}\right] = \qquad (3.8)$$

$$= \alpha'' \exp\left[-\frac{1}{2}\left\{\left(\frac{n}{\sigma^2}+\frac{1}{\sigma_0^2}\right)\mu^2 - 2\left(\frac{1}{\sigma^2}\sum_{k=1}^{n}x_k + \frac{\mu_0}{\sigma_0^2}\right)\mu\right\}\right].$$

В выражении (3.8) множители, не зависящие от $\mu$, включены в константы $\alpha'$ и $\alpha''$.

Из (3.8) также следует, что плотность $p(\mu/X)$ является нормальной. Обозначим параметры полученной нормальной плотности, как $\mu_n$ и $\sigma_n^2$, которые могут быть получены приравниваем коэффициентов из выражения (3.8) соответствующим коэффициентам из выражения

$$p(\mu / Y) = \frac{1}{\sqrt{2\pi}\sigma_n} \exp\left[-\frac{1}{2}\left(\frac{\mu-\mu_n}{\sigma_n}\right)^2\right].$$

Отсюда получаем

$$\frac{1}{\sigma_n^2} = \frac{n}{\sigma^2} + \frac{1}{\sigma_0^2}$$

и

$$\frac{\mu_n}{\sigma_n^2} = \frac{\sum_{k=1}^{n} y_k}{\sigma^2} + \frac{\mu_0}{\sigma_0^2}.$$

Решая уравнения в явном виде, получаем

$$\mu_n = \frac{n\sigma_0^2}{n\sigma_0^2 + \sigma^2} m_n + \frac{\sigma^2}{n\sigma_0^2 + \sigma^2}\mu_0, \qquad (3.9)$$

где $m_n$ – выборочное среднее,

$$m_n = \frac{1}{n}\sum_{k=1}^{n} x_k ,$$

$$\sigma_n^2 = \frac{\sigma_0^2\sigma^2}{n\sigma_0^2 + \sigma^2}$$

$\sigma_n^2$ – выборочная дисперсия.

Из уравнения (3.9) видно, что оптимальная оценка математического ожидания апостериорной плотности представляет собой линейную комбинацию априорного математического ожидания $\mu_0$ и выборочного среднего $m_n$, полученного по $n$ выборкам. Коэффициенты перед этими величинами неотрицательны и в сумме равны единице.

Если достоверность априорных данных очень высока ($\sigma_0^2 \cong 0$), то в качестве оценки следует использовать априорное математическое ожидание $\mu_0$. Это говорит о том, что никакое число измерений не может поколебать нас в уверенности в априорных данных.

При альтернативной ситуации ($\sigma_0^2 \neq 0$, а число наблюдений стремится к бесконечности: $n \to \infty$, $\frac{\sigma^2}{n} \to 0$) в качестве оценки следует использовать выборочное среднее $m_n$, не доверяя, в свою очередь, априорным сведениям. Вообще относительный баланс между априорными представлениями и опытными данными определяется отношением $\sigma^2$ к $\sigma_0^2$, называемым иногда догматизмом. Если догматизм не бесконечен, то после получения достаточного

числа выборок априорные данные перестают играть сколько-нибудь заметную роль, а оценка стремится к выборочному среднему.

После получения апостериорной плотности $p(\mu/X)$ остается определить «условную по классу» плотность $p(x/X)$ (в данном выводе мы полагаем, что все выборки принадлежат одному, например, $j$-му классу с априорной плотностью $p(Hj)$). Для определения требуемой плотности вычислим

$$p(x/X)=\int p(x/\mu)p(\mu/X)d\mu=$$

$$\int\frac{1}{\sqrt{2\pi}\sigma}\exp\left[-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right]\frac{1}{\sqrt{2\pi}\sigma_n}\exp\left[-\frac{1}{2}\left(\frac{\mu-\mu_n}{\sigma_n}\right)^2\right]d\mu=$$

$$=f(\sigma,\sigma_n)\exp\left[-\frac{1}{2}\frac{(x-\mu_n)^2}{\sigma^2+\sigma_b^2}\right].$$

Отсюда следует, что плотность $p(\overline{x}/\boldsymbol{X})$ является нормальной со средним значением, равным $\mu_n$, и дисперсией $\sigma^2+\sigma_0^2$.

Для получения байесовского решения полученную плотность для каждого $j$-го класса следует умножить на априорную вероятность этого класса и выбрать максимальное значение

$$\hat{j}_{опт.}=\max_j p(H_j/x,X_j)=\max_j\frac{p(x/H_i,X_i)P(H_i)}{\sum_{j=1}^{M}p(x/H_j,X_j)P(H_j)}.$$

### *3.3. Эффективность оценки. Нижняя граница дисперсии несмещенной оценки. Неравенство Крамера-Рао*

Предположим, что мы производим оценку неизвестного неслучайного параметра $\Theta$ и в результате измерений получаем так называемую несмещенную оценку, т. е. такую оценку, математическое ожидание которой равняется значению самого оцениваемого параметра, т. е. $M[\hat{\Theta}] = \Theta$. Для того чтобы определить качество оценки, следует определить ее дисперсию, которая вычисляется следующим образом:

$$\sigma^2[\hat{\Theta}(x)-\Theta]=M\left\{\left[\hat{\Theta}(x)-\Theta\right]^2\right\}.$$

Дисперсия дает меру рассеяния ошибки. Наилучшей оценкой была бы, по-видимому, несмещенная оценка с минимальной дисперсией. Однако регулярной процедуры, которая бы приводила к получению алгоритма, формирующего несмещенную оценку с минимально возможной дисперсией, не существует.

В этой ситуации имеет смысл получить выражение для нижней границы дисперсии любой несмещенной оценки. Знание границы позволит сравнить дисперсию той или иной оценки с этой границей, и в том случае, если будет получено совпадение дисперсии оценки с нижней границей, может быть сделан вывод, что мы получили наилучшую оценку. Если же точное совпадение не обеспечено, то и в этом случае мы можем судить, насколько наша оценка отличается от потенциально достижимой.

Докажем следующее утверждение. Если $\hat{\Theta}(x)$ – любая несмещенная оценка величины $\Theta$, то

$$\sigma^2\left[\hat{\Theta}(x)-\Theta\right] \geq \left(M\left\{\left[\frac{\partial \ln p(x/\Theta)}{\partial\Theta}\right]^2\right\}\right)^{-1} \qquad (3.10)$$

или, что эквивалентно,

$$\sigma^2\left[\hat{\Theta}(x)-\Theta\right] \geq \left\{-M\left[\frac{\partial^2 \ln p(x/\Theta)}{\partial\Theta^2}\right]\right\}^{-1} . \qquad (3.11)$$

При этом мы считаем, что производные

$$\frac{\partial \ln p(x/\Theta)}{\partial\Theta} \quad \text{и} \quad \frac{\partial^2 \ln p(x/\Theta)}{\partial\Theta^2}$$

существуют и являются абсолютно интегрируемы.

Неравенства (3.10) и (3.11) обычно называются границами Крамера-Рао. Любая оценка, удовлетворяющая указанной границе со знаком равенства, называется эффективной оценкой.

Доказательство этого положения основано на использовании неравенства Буняковского-Шварца. Так как по нашему предположению оценка считается несмещенной, запишем

$$M\left[\hat{\Theta}(x)-\Theta\right]=\int_{-\infty}^{\infty}p(x/\Theta)\left[\hat{\Theta}(x)-\Theta\right]dx=0\,. \qquad (3.25)$$

Дифференцируя обе части по Θ, имеем

$$\frac{d}{d\Theta}\int_{-\infty}^{\infty}p(x/\Theta)\left[\hat{\Theta}(x)-\Theta\right]\,dx=\int_{-\infty}^{\infty}\frac{d}{d\Theta}\left\{p(x/\Theta)\left[\hat{\Theta}(x)-\Theta\right]\right\}dx\;= \qquad (3.12)$$
$$=-\int_{-\infty}^{\infty}p(x/\Theta)\,dx\;+\int_{-\infty}^{\infty}\frac{dp(x/\Theta)}{d\Theta}\left[\hat{\Theta}(x)-\Theta\right]dx=0.$$

Первый интеграл равен 1. Кроме того, заметим, что

$$\frac{dp(x/\Theta)}{d\Theta}=\frac{d\ln p(x/\Theta)}{d\Theta}\,p(x/\Theta)\,. \qquad (3.13)$$

Подставляя (3.13) в (3.12), получаем

$$\int_{-\infty}^{\infty}\frac{d\ln p(x/\Theta)}{d\Theta}\,p(x/\Theta)\left[\hat{\Theta}(x)\;-\;\Theta\right]dx=1.$$

Перепишем подынтегральное выражение в следующем виде:

$$\int_{-\infty}^{\infty}\left[\frac{d\ln p(x/\Theta)}{d\Theta}\sqrt{p(x/\Theta)}\right]\left[\sqrt{p(x/\Theta)}\left[\hat{\Theta}(x)\;-\;\Theta\right]\right]dx=1.$$

и используем неравенство Буняковского-Шварца:

$$\int_{-\infty}^{\infty}\left[\frac{d\ln p(x/\Theta)}{d\Theta}\right]^2p(x/\Theta)\,dx\times\int_{-\infty}^{\infty}\left[\hat{\Theta}(x)\;-\;\Theta\right]^2p(x/\Theta)\;dx\geq 1. \qquad (3.14)$$

В связи с тем, что каждый из сомножителей представляет собой математическое ожидание, имеем следующее неравенство:

$$M\left\{\left[\hat{\Theta}(x)\;-\;\Theta\right]^2\right\}\geq\left\{M\left[\frac{d\ln p(x/\Theta)}{d\Theta}\right]^2\right\}^{-1}.$$

Итак, неравенство (3.10) можно считать доказанным. Для доказательства неравенства (3.11) заметим, что

$$\int_{-\infty}^{\infty}p(x/\Theta)dx=1.$$

Дифференцируя по Θ, имеем

$$\int_{-\infty}^{\infty}\frac{\partial p(x/\Theta)}{\partial\Theta}dx=\int_{-\infty}^{\infty}\frac{\partial\ln p(x/\Theta)}{\partial\Theta}\,p(x/\Theta)dx\,. \qquad (3.15)$$

Вновь дифференцируя по Θ и применяя (3.13), получим

$$\int_{-\infty}^{\infty} \frac{\partial^2 \ln p(x/\Theta)}{\partial \Theta^2} p(x/\Theta) dx + \int_{-\infty}^{\infty} \left( \frac{\partial \ln p(x/\Theta)}{\partial \Theta} \right)^2 p(x/\Theta) dx = 0.$$

Отсюда следует

$$M\left[ \frac{\partial^2 \ln p(x/\Theta)}{\partial \Theta^2} \right] = -M\left[ \frac{\partial \ln p(x/\Theta)}{\partial \Theta} \right]^2.$$

Последнее равенство означает справедливость условия (3.11).

***Неравенство Крамера-Рао позволяет сделать ряд важных замечаний.***

1. Любая несмещенная оценка имеет дисперсию больше, чем некоторое число.

2. Неравенство Буняковского-Шварца (3.14) выполняется тогда и только тогда, когда

$$\frac{\partial \ln p(x/\Theta)}{\partial \Theta} = \left( \hat{\Theta}(x) - \Theta \right) k(\Theta). \qquad (3.16)$$

Если эффективная оценка существует (равенство (3.16) выполняется), то эта оценка является оценкой максимального правдоподобия. Действительно, уравнение правдоподобия имеет вид

$$\left. \frac{\partial \ln p(x/\Theta)}{\partial \Theta} \right|_{\Theta = \hat{\Theta}_{мп}} = 0.$$

Для того чтобы правая часть равенства (3.16) принимала нулевое значение, оценка $\hat{\Theta}$ должна быть равна $\hat{\Theta}_{мп}$.

3. Если эффективной оценки не существует (равенство (3.16) не выполняется), то неизвестно, насколько оптимальной является оценка максимального правдоподобия (насколько близко она приближается к границе). В этой ситуации границу и дисперсию оценки приходится вычислять и полученные величины сравнивать. Однако достаточно обнадеживающим является тот факт, что оценка по максимуму правдоподобия является асимптотически эффективной, иначе говоря, при стремлении размера выборки (размерности вектора $x$ к бесконечности) дисперсия оценки максимального правдоподобия стремится к своей границе.

# 4. Непараметрические методы

В предыдущей главе мы рассматривали вопросы обучения с учителем, допуская, что вид основных плотностей распределения известен. Для многих случаев принятия решения это допущение неверно. Далеко не всегда распространенные параметрические формы полностью соответствуют плотностям распределения, встречающимся на практике. В данном разделе мы рассмотрим непараметрические процедуры, которыми можно пользоваться, не считая, что вид основных плотностей распределения известен.

Для распознавания образов интерес представляют несколько различных видов непараметрических методов. Один из методов состоит из процедур оценки плотности распределения $p(\bar{x}/H_i)$ на основании выбранных образов. Если эти оценки удовлетворительны, то при проектировании оптимального классификатора ими можно заменить истинные значения плотности распределения.

Другой метод состоит из процедур прямой оценки апостериорных вероятностей $P(H_i/\bar{x})$. Этот метод близок к таким решающим процедурам, как правило «ближайшего соседа», в котором, обходя вероятностные оценки, сразу переходят к решающим функциям.

И наконец, есть непараметрические процедуры, преобразующие пространство признаков так, чтобы в преобразованном пространстве можно было использовать параметрические методы.

## *4.1. Оценка плотности распределения*

Методы оценки неизвестной плотности распределения основываются на том, что вероятность $P$ попадания вектора $x$ в область $R$ задается выражением:

$$P=\int_R p(x')dx' . \qquad (4.1)$$

Таким образом, $P$ есть усредненный вариант плотности распределения $p(x)$, и можно оценить это значение p посредством оценки вероятности P. Предположим, что n выборок $x_1, x_2, \ldots, x_n$ берутся независимо друг от друга в

соответствии с вероятностным законом $p(x)$. Очевидно, что вероятность попадания k из n выборок в R задается биномиальным законом

$$P_k = \binom{n}{k} P^k (1-P)^{n-k} \tag{4.2}$$

и ожидаемой величиной $k$ будет

$$E[k] = nP. \tag{4.3}$$

Более того, это биномиальное распределение для $k$ имеет довольно резко выраженные максимумы около среднего значения, поэтому мы считаем, что отношение $k/n$ будет хорошей оценкой вероятности $P$, а следовательно, и сглаженной плотности распределения. Если теперь мы допустим, что $p(x)$ непрерывна и область R настолько мала, что p в ее пределах меняется незначительно, то можем написать

$$\int_R p(x')dx' \approx p(x)V, \tag{4.4}$$

где $x$ — это точка внутри $R$ и $V$ — объем $R$. Объединяя уравнения (4.1)–(4.4), получаем следующую очевидную оценку для $p(x)$:

$$p(x) \approx \frac{k/n}{V},$$

Остается решить несколько проблем практического и теоретического плана. Если мы фиксируем объем $V$ и делаем все больше и больше выборок, отношение $k/n$ сойдется (по вероятности) требуемым образом, но при этом мы получаем оценку только пространственно усредненной величины $p(x)$:

$$\frac{p}{V} = \frac{\int_R p(x')dx'}{\int_R dx'}.$$

Если мы хотим получить $p(x)$, а не усредненный ее вариант, необходимо устремить $V$ к нулю. Однако если зафиксировать количество n выборок и позволить $V$ стремиться к нулю, то область в конечном счете станет настолько мала, что не будет содержать в себе никаких выборок, и наша оценка $p(x) \cong 0$ будет бесполезной.

С практической точки зрения количество выборок всегда ограничено, так что нельзя позволить объему $V$ становиться бесконечно малым. Если

приходится пользоваться таким видом оценки, то нужно допускать определенную дисперсию отношения $k/n$ и определенное усреднение плотности распределения $p(x)$.

С теоретической точки зрения интересно, как можно обойти эти ограничения при наличии неограниченного количества выборок. Предположим, что мы пользуемся следующей процедурой. Для оценки плотности распределения $x$ мы образуем последовательность областей, содержащих $x$. Первая область будет соответствовать одной выборке, вторая — двум и т. д. Пусть $V_n$ будет иметь объем $R_n$, $k_n$— количество выборок, попадающих в $R_n$, а $p_n(x)$ — $n$-я оценка $p(x)$:

$$p_n(x)=\frac{k_m/n}{V_n}. \qquad (4.5)$$

Если $p_n(x)$ должна сойтись к $p(x)$, то следует выполнить три условия:

$$1)\ \lim_{n\to\infty} V_n = 0;$$
$$2)\ \lim_{n\to\infty} k_n = \infty;$$
$$3)\ \lim_{n\to\infty} k_n/n = 0.$$

Первое условие обеспечивает сходимость пространственно усредненного $P/V$ к $p(x)$ при однородном сокращении областей и при непрерывности $p$ в $x$. Второе условие, имеющее смысл только при $p(x) \neq 0$, обеспечивает сходимость (по вероятности) отношения частот к вероятности $P$.

Совершенно ясно, что третье условие необходимо, если $p_n(x)$, заданная соотношением (4.5), вообще должна сойтись. Это условие говорит также о том, что, хотя в конечном счете в небольшую область $R_n$ попадает огромное количество выборок, оно составит лишь незначительно малую часть всего количества выборок.

Существует два общих способа получения последовательностей областей, удовлетворяющих указанным выше условиям. Первый способ заключается в сжатии начальной области за счет определения объема $V_n$, как некоторой функции от $n$, такой, чтобы $V_n = 1/\sqrt{n}$. Далее следует показать, что случайные величины $k_n$ и $k_n/n$ ведут себя правильно или, что $p_n(x)$ сходится к $p(x)$. В этом

заключается метод парзеновского окна, рассматриваемый в следующем разделе. Во втором методе $k_n$ определяется как некоторая функция от $n$: $k_n = \sqrt{n}$. Здесь объем $V_n$ увеличивается до тех пор, пока не охватит $k_n$ «соседей» $x$. Это метод оценки по $k_n$ ближайшим соседям. Оба эти метода действительно обеспечивают сходимость, хотя трудно сказать что-либо определенное об их поведении при конечном числе выборок.

### 4.1.1. Парзеновские окна

Зададим область $R_n$ в виде $d$-мерного гиперкуба, длина ребра которого равна $h_n$ и, следовательно, объем равен

$$V_n = h_n^d.$$

Определим функцию окна:

$$\varphi(u) = \begin{cases} 1, если \left|u_j\right| \le 1/2,\ j = 1,2,.....d \\ 0\ в\ противном\ случае \end{cases}.$$

Таким образом, $\varphi(u)$ определяет единичный куб с центром в начале координат. Отсюда следует, что $\varphi\left(\frac{x - x_i}{h_n}\right)$ равняется единице, если $x_i$ находится в гиперкубе объема $V_n$ с центром в $x$, или нулю в любом другом случае. Следовательно, количество выборок в этом гиперкубе задается выражением:

$$k_n = \sum_{i=1}^{n} \varphi\left(\frac{x - x_i}{h_n}\right)$$

Подставляя полученное выражение в (4.5), получаем оценку плотности вероятности:

$$p_n(x) = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{V_n} \varphi\left(\frac{x - x_i}{h_n}\right). \qquad (4.6)$$

Полученная оценка должна быть неотрицательна с площадью, равной 1. Для этого необходимо:

$$\varphi(u) \ge 0;$$
$$\int \varphi(u)\,du = 1.$$

Достоинство рассмотренного метода заключается в том, что при достаточно большом числе выборок оценка плотности вероятности сходится к

неизвестной плотности. В то же время требуемое число выборок может оказаться чрезвычайно большим. Это число может быть оказаться слишком велико для реальной ситуации, причем практически отсутствуют способы уменьшения требуемого объема данных. Более того, потребность в числе выборок растет экспоненциально с увеличением размерности пространства признаков.

### 4.1.2. Оценка методом $k_n$ ближайших соседей

Одна из проблем, с которой сталкиваются при использовании метода парзеновского окна, заключается в выборе последовательности объемов ячеек $V_1$, $V_2$ ... Если $V_1$ слишком мал, то большинство объемов будут пустыми, и оценка $p_n(x)$ будет ошибочной. Если $V_1$ слишком велик, то из-за усреднения по объему ячейки могут быть потеряны важные пространственные отклонения от $p(x)$. Кроме того, вполне может случиться, что объем ячейки, уместный для одного значения **x**, может совершенно не годиться для других случаев.

Один из возможных способов решения этой проблемы — сделать объем ячейки функцией данных, а не количества выборок. Например, чтобы оценить $p(x)$ на основании n выборок, можно центрировать ячейку вокруг **x** и позволить ей расти до тех пор, пока она не вместит $k_n$ выборок, где $k_n$ есть некая определенная функция от $n$. Эти выборки будут $k_n$ ближайшими соседями $x$. Если плотность распределения вблизи $x$ высокая, то ячейка будет относительно небольшой, что приводит к хорошему разрешению. Если плотность распределения невысокая, то ячейка возрастает, но рост приостанавливается вскоре после ее вступления в области более высокой плотности распределения. В любом случае, если мы берем

$$p_n(x)=\frac{k_n/n}{V_n}, \qquad (4.7)$$

мы хотим, чтобы $k_n$ стремилось к бесконечности при стремлении $n$ к бесконечности, т.к. этот факт гарантирует, что $k_n/n$ будет хорошей оценкой вероятности попадания точки в ячейку объема $V_n$. Однако мы хотим также, чтобы $k_n$ росло достаточно медленно для того, чтобы размер ячейки,

необходимый для вмещения $k_n$ выборок, сжался до нуля. Таким образом, из формулы (4.7) видно, что отношение $k_n/n$ должно стремиться к нулю.

Можно показать, что условия

$$\lim_{n\to\infty} k_n = \infty \; и \; \lim_{n\to\infty} k_n / n = 0$$

являются необходимыми и достаточными для сходимости $p_n(x)$ и $p(x)$ по вероятности во всех точках, где плотность $p$ непрерывна.

## 4.2. Оценка апостериорных вероятностей. Правило ближайших соседей

Методы, рассмотренные в п.4.1, могут быть с успехом использованы для оценки апостериорных вероятностей $P(H_i /x)$ и, следовательно, для принятия решений по критерию максимума названных вероятностей.

Предположим, что мы размещаем ячейку объема $V$ вокруг $x$ и захватываем $k$ обучающих выборок, $k_i$ из которых оказываются принадлежащими гипотезе $H_i$. При этом оценкой совместной вероятности $p(x, H_i)$ будет

$$p_n(x, H_i) = \frac{k_i / n}{V}.$$

Таким образом, приемлемой оценкой для $P(H_i/x)$ будет

$$P_n(H_i / x) = \frac{p_n(x, H_i)}{\sum_{j=1}^{M} p_n(x, H_i)} = \frac{k_i}{k}. \qquad (4.8)$$

Из формулы (4.8) следует, что оценкой апостериорной вероятности того факта, что справедлива гипотеза $H_i$, является доля выборок, отмеченных при обучении как принадлежащих $i$-му классу. Если число обучающих выборок достаточно велико, а размер ячейки достаточно мал, то получаем практически наилучшую оценку апостериорной плотности.

Интересно, что хорошее решение (близкое к оптимальному) получается при принятии решения о справедливости той или иной гипотезы всего по одному ближайшему соседу. Смысл данного правила заключается в том, что

решение принимается в пользу той гипотезы, которой соответствует самая близкая[1] обучающая выборка.

Очевидным расширением правила ближайшего соседа является правило $k$ ближайших соседей. В этом случае анализируют $k$ обучающих выборок, расположенных рядом с наблюдаемой выборкой. Предпочтение отдается той гипотезе, которая порождает большее число обучающих выборок в окрестности наблюдаемой выборки, для которой принимается решение.

### *4.3. Аппроксимации путем разложения в ряд*

Все описанные до сих пор непараметрические методы имеют тот недостаток, что требуют хранения в памяти всех выборок. А так как для получения хороших оценок необходимо большое количество выборок, то потребность в памяти может быть слишком велика. Кроме того, может потребоваться значительное время вычисления каждый раз, когда один из методов используется для оценки величины $p(x)$ или классификации нового $x$. При определенных обстоятельствах процедуру окна Парзена можно несколько видоизменить, чтобы значительно сократить эти требования.

Основная идея заключается в аппроксимации функции окна путем разложения ее в конечный ряд, что делается с приемлемой точностью в представляющей интерес области. Если нам сопутствует удача и мы можем найти два множества функций $\varphi_j(x)$ *и* $\varphi_j(x)$, которые допускают разложение

$$\varphi\left(\frac{x-x_j}{h_n}\right)=\sum_{j=1}^{m} a_j \phi_j(x)\chi_j(x_i), \qquad (4.9)$$

тогда

$$\sum_{i=1}^{n}\varphi\left(\frac{x-x_i}{h_n}\right)=\sum_{j=1}^{m} a_j \phi_j(x)\sum_{i=1}^{n}\chi_j(x_i),$$

и из уравнения (4.11) имеем

$$p_n(x) = \sum_{j=1}^{m} b_j \phi_j(x),$$

где

[1] Например, по среднеквадратичному расстоянию.

$$b_j = \frac{a_j}{nV_n} \sum_{i=1}^{n} \chi_j(x_i) . \qquad (4.10)$$

Этот подход имеет некоторые преимущества в том случае, когда можно получить достаточно точное разложение с приемлемым значением $m$. Информация, содержащаяся в $n$ выборках, сводится к m коэффициентам $b_j$. Если получают дополнительные выборки, соотношение (4.10) для $b_j$ можно легко обновить, причем количество коэффициентов остается неизменным.

Если функции $\psi_j$ и $\chi_j$ являются полиномами от компонент $x$ и $x_i$, то выражение для оценки $p_n(x)$ есть также полином, который можно довольно эффективно вычислить. Более того, использование этой оценки для получения разделяющих функций $p(x/H_i)$, $P(H_i)$ приводит к простому способу получения полиномиальных разделяющих функций.

Следует отметить одну из проблем, возникающую при применении этого способа. Основным достоинством функции окна является ее тенденция к возрастанию в начале координат и снижению в других точках, так что $\varphi((x-x_i)/h_n))$ будет иметь резкий максимум при $x = x_i$ и мало влиять на аппроксимацию $p_n(x)$ для удаленного от $x_i$. К сожалению, полиномы обладают досадным свойством, заключающимся в том, что они могут содержать неограниченное количество членов. Поэтому при разложении полинома могут обнаружиться члены, ассоциируемые с $x_i$, удаленным от $x$, сильно влияющим на разложение. Следовательно, важно убедиться, что разложение каждой функции окна действительно точное в представляющей интерес области, а для этого может потребоваться большое число членов.

Существует много видов разложения в ряд. Читатели, знакомые с интегральными уравнениями, вполне естественно интерпретируют соотношение (4.9) как разложение ядра $\varphi(x, x_i)$ в ряды по собственным функциям. Вместо вычисления собственных функций можно выбрать любое приемлемое множество функций, ортогональных в интересующей нас области, и получить согласие по методу наименьших квадратов с функцией окна. Мы применим еще более непосредственный подход и разложим функцию окна в

ряд Тейлора. Для простоты ограничимся одномерным случаем с гауссовской функцией окна:

$$\sqrt{\pi}\varphi(u) = e^{-u^2} \approx \sum_{j=0}^{m-1}(-1)^j \frac{u^{2j}}{j!}.$$

Самым точным это разложение будет в окрестности $u = 0$, где ошибка будет составлять менее $u^{2m}/m!$. Если мы подставим $u = \frac{x - x_j}{h}$, то получим полином степени $2(m-1)$ от $x$ и $x_i$. Например, если $m = 2$, то

$$\sqrt{\pi}\varphi\left(\frac{x - x_i}{h}\right) \approx 1 - \left(\frac{x - x_i}{h}\right)^2 = 1 + \frac{2}{h^2}xx_i - \frac{1}{h^2}x^2 - \frac{1}{h^2}x_i^2$$

и, таким образом,

$$\sqrt{\pi}p_n(x) = \frac{1}{nh}\sum_{i=1}^{n}\sqrt{\pi}\varphi\left(\frac{x - x_i}{h}\right) \approx b_0 + b_1 x + b^2 x^2,$$

где

$$b_0 = \frac{1}{h} - \frac{1}{h^2 n}\sum_{i=1}^{n} x_i^2,$$

$$b_1 = \frac{2}{h^2 n}\sum_{i=1}^{n} x_i,$$

$$b_2 = -\frac{1}{h^2}.$$

Это простое разложение сжимает информацию из $n$ выборок в три коэффициента $b_0, b_1, b_2$. Оно будет точным, если наибольшее значение $|x - x_i|$ не превышает $h$. К сожалению, это заставляет нас пользоваться очень широким окном, которое не дает большого разрешения. Беря большое количество членов, мы можем использовать более узкое окно. Если мы считаем r наибольшим значением $|x - x_i|$, то, пользуясь тем фактом, что ошибка в m-членном разложении функции $\sqrt{\pi}\varphi[(x - x_i)/h]$ меньше, чем $(r/h)^{2m}m!$, и пользуясь аппроксимацией Стирлинга для m!, найдем, что ошибка в аппроксимации $p_n(x)$ будет меньше, чем

$$\frac{1}{\sqrt{\pi}h}\frac{\left(\frac{r}{h}\right)^{2m}}{m!} \approx \frac{1}{\sqrt{\pi}h\sqrt{2\pi m}}\left[\left(\frac{e}{m}\right)\left(\frac{r}{h}\right)^2\right]^m$$

Таким образом, ошибка мала только тогда, когда $m > e(r/h)^2$. Это говорит о том, что требуется много членов, если ширина окна $h$ невелика по сравнению с расстоянием $r$ от $x$ до наиболее удаленной выборки.

### *4.4. Линейный дискриминант Фишера*

Одна из ключевых проблем, встречающихся при распознавании образов, заключается в так называемом «проклятии размерности». Процедуры, выполняемые аналитически в пространствах признаков небольшой размерности, могут стать совершенно неприменимыми в пространствах с числом признаков, равным нескольким десяткам. Были разработаны различные методы уменьшения размерности пространства признаков в надежде получить задачу, поддающуюся решению.

Можно уменьшить размерность с $d$ измерений до одного, просто проецируя $d$-мерные данные на прямую. Если выборки образуют хорошо разделенные компактные группы в $d$-пространстве, проекция на произвольную прямую обычно смешивает выборки из всех классов. Однако, вращая эту прямую, мы можем найти такую ориентацию, при которой спроецированные выборки будут хорошо разделены. Именно это является целью классического дискриминантного анализа.

Предположим, что мы имеем множество $n$ $d$-мерных выборок $x_1,\ldots, x_n$, из которых $n_1$ выборок относятся к подмножеству $X_1$, соответствующему гипотезе $H_1$, и $n_2$ выборок относятся к подмножеству $X_2$, соответствующему гипотезе $H_2$. Если мы образуем линейную комбинацию компонент вектора $x$, получим скалярную величину

$$y = \overline{w}^t \overline{x}$$

и соответствующее множество n выборок $y_1,\ldots, y_n$, разделенное на подмножества $Y_1$ и $Y_2$. Геометрически, если $||\overline{w}|| = 1$, каждая компонента $y_i$ есть проекция соответствующего $x_i$ на прямую в направлении $\overline{w}$. В действительности величина $\overline{w}$ не имеет реального значения, поскольку она просто определяет масштаб $y$. Однако направление $\overline{w}$ имеет значение. Если мы

вообразим, что выборки, помеченные $H_1$, попадают более или менее в одну группу, а выборки, помеченные $H_2$, попадают в другую, то мы хотим, чтобы проекции на прямой были хорошо разделены и не очень перемешаны. На рис. 4.1 показан выбор двух различных значений $w$ для двумерного случая.

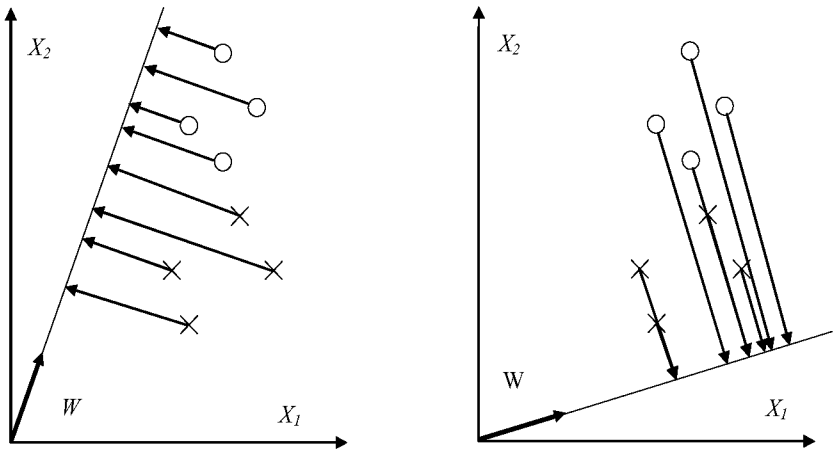



Рис. 4.1

Мерой разделения спроецированных точек служит разность средних значений выборки. Если $m_i$ есть среднее значение $d$-мерной выборки, заданное как

$$m_i = \frac{1}{n_i} \sum_{x \in X_i} x,$$

то среднее значение выборки для спроецированных точек задается посредством

$$\tilde{m}_i = \frac{1}{n_i} \sum_{y \in Y_i} y = \frac{1}{n_i} \sum_{y \in Y_i} \overline{w}^t \overline{x}.$$

Отсюда следует, что $\left|\tilde{m}_1 - \tilde{m}_2\right| = \left|\overline{w}^t \left(\overline{m}_1 - \overline{m}_2\right)\right|$ и что мы можем сделать эту разность сколь угодно большой, масштабируя $\overline{w}$.

Чтобы получить хорошее разделение спроецированных данных, необходимо, чтобы разность между средними значениями была велика относительно некоторого показателя стандартных отклонений для каждого

класса. Вместо получения дисперсий выборок определим разброс для спроецированных выборок, помеченных $H_i$, посредством

$$\tilde{s}_i^2 = \sum_{y \in Y_i} (y - \tilde{m}_i)^2 .$$

Таким образом, $(1/n)(\tilde{s}_1^2 + \tilde{s}_2^2)$ является оценкой дисперсии совокупности данных, где $(\tilde{s}_1^2 + \tilde{s}_2^2)$ называется *полным разбросом внутри класса спроецированных выборок*. Линейный дискриминант Фишера тогда определяется как такая линейная разделяющая функция $w^t x$, для которой функция критерия

$$J(w) = \frac{|\tilde{m}_1 - \tilde{m}_2|}{\tilde{s}_1^2 + \tilde{s}_2^2}$$

максимальна.

Чтобы получить $J$ как явную функцию от $\boldsymbol{w}$, определим матрицы разброса $S_i$ и $S_w$ посредством

$$S_i = \sum_{x \in X_i} (\bar{x} - \bar{m}_i)(\bar{x} - \bar{m}_i)^t$$

и

$$S_W = S_1 + S_2 .$$

Тогда

$$\tilde{s}_i^2 = \sum_{x \in X_i} \left( \bar{w}^t \bar{x} - \bar{w}^t \bar{m}_i \right)^2 = \sum_{x \in X_i} \bar{w}^t (\bar{x} - \bar{m}_i)(\bar{x} - \bar{m}_i)^t \bar{w} = \bar{w}^t S_i \bar{w} .$$

Отсюда

$$\tilde{s}_1^2 + \tilde{s}_2^2 = \bar{w}^t S_W \bar{w} .$$

Аналогично

$$(\tilde{m}_1 - \tilde{m}_2)^2 = \left( \bar{w}^t \bar{m}_1 - \bar{w}^t \bar{m}_2 \right)^2 = \bar{w}^t (\bar{m}_1 - \bar{m}_2)(\bar{m}_1 - \bar{m}_2)^t \bar{w} = \bar{w}^t S_B \bar{w} ,$$

где

$$S_B = (\bar{m}_1 - \bar{m}_2)(\bar{m}_1 - \bar{m}_2)^t .$$

Матрица $S_w$ называется матрицей разброса внутри класса. Она пропорциональна ковариационной выборочной матрице для совокупности $d$-мерных данных. Она будет симметричной, положительно полуопределенной

и, как правило, невырожденной, если $n > d$. $S_B$ называется матрицей разброса между классами. Она также симметричная и положительно полуопределенная, но из-за того, что она является внешним произведением двух векторов, ее ранг будет не больше 1. В частности, для любого $\overline{w}$ направление $S_B\overline{w}$ совпадает с направлением $\overline{m}_1 - \overline{m}_2$ и $S_B$ – вполне вырожденная матрица.

При помощи $S_B$ и $S_w$ функцию критерия $J$ можно представить в виде

$$J(w) = \frac{\overline{w}^t S_B \overline{w}}{\overline{w}^t S_W \overline{w}}.$$

Это выражение хорошо известно в математической физике как обобщенное частное Релея. Легко показать, что вектор $\overline{w}$, который максимизирует $J$, должен удовлетворять соотношению

$$S_B\overline{w} = \lambda S_W \overline{w},$$

что является обобщенной задачей определения собственного значения.

Если $S_W$ является невырожденной, мы можем получить обычную задачу определения собственного значения, написав

$$S_W^{-1} S_B \overline{w} = \lambda \overline{w}.$$

В нашем частном случае не нужно находить собственные значения и собственные векторы $S_W^{-1}S_B$ из-за того, что направление $S_B\overline{w}$ всегда совпадает с направлением $\overline{m}_1 - \overline{m}_2$. Поскольку масштабный множитель для w несуществен, мы можем сразу написать решение

$$\overline{w} = S_W^{-1}(\overline{m}_1 - \overline{m}_2). \qquad (4.11)$$

Таким образом, мы получили линейный дискриминант Фишера - линейную функцию с максимальным отношением разброса между классами к разбросу внутри класса. Задача была преобразована из $d$-мерной в более приемлемую одномерную. Это отображение n-мерного множества на одномерное, и теоретически оно не может уменьшить минимально достижимый уровень ошибки.

Когда условные плотности распределения $p(\overline{x}/H_i)$ являются многомерными нормальными с равными ковариационными матрицами $\Sigma$, то

даже не нужно ничем жертвовать. В этом случае мы вспоминаем, что граница оптимальных решений удовлетворяет уравнению

$$\overline{w}^t \overline{x} + w_0 = 0,$$

где

$$\overline{w} = \Sigma^{-1} (\overline{\quad} - \overline{\quad});$$

и $w_0$ есть константа, включающая в себя $w$ и априорные вероятности.

Если мы используем средние значения и ковариационную матрицу выборок для оценки $\mu_i$ и $\Sigma$, то получаем вектор в том же направлении, что и $\overline{w}$, удовлетворяющий (4.11), который максимизирует $J$. Таким образом, для нормального случая с равными ковариациями оптимальным решающим правилом будет решение $H_1$, если линейный дискриминант Фишера превышает некоторое пороговое значение, и решение $H_2$ — в противном случае.

### *4.5. Множественный дискриминантный анализ*

Для задачи с M классами естественное обобщение линейного дискриминанта Фишера включает *(M – 1)* разделяющих функций.

Таким образом, проекция будет из $d$-мерного пространства на *(M—1)*-мерное пространство, причем принимается, что $d \geq M$. Обобщение для матрицы разброса внутри класса очевидное:

$$S_W = \sum_{i=1}^{M} S_i,$$

где, как и прежде,

$$S_i = \sum_{x \in X_i} (\overline{x} - \overline{m}_i)(\overline{x} - \overline{m}_i)^t,$$

и

$$m_i = \frac{1}{n_i} \sum_{x \in X_i} x.$$

Соответствующее обобщение для $S_B$ не так очевидно. Предположим, что определяем полный вектор средних значений $m$ и полную матрицу разброса $S_T$ посредством

$$m = \frac{1}{n}\sum_{x} x = \frac{1}{n}\sum_{i=1}^{M} n_i m_i,$$

и

$$S_T = \sum_{x} (\bar{x} - \bar{m})(\bar{x} - \bar{m})^t.$$

Отсюда следует, что

$$S_T = \sum_{i=1}^{M}\sum_{x\in X_i} (\bar{x} - \bar{m}_i + \bar{m}_i - \bar{m})(\bar{x} - \bar{m}_i + \bar{m}_i - \bar{m})^t =$$

$$= \sum_{i=1}^{M}\sum_{x\in X_i} (\bar{x} - \bar{m}_i)(\bar{x} - \bar{m}_i)^t + \sum_{i=1}^{M}\sum_{x\in X_i} (\bar{m}_i - \bar{m})(\bar{m}_i - \bar{m})^t =$$

$$= S_W + \sum_{i=1}^{M} n_i (\bar{m}_i - \bar{m})(\bar{m}_i - \bar{m})^t.$$

Естественно определять этот второй член как матрицу разброса между классами, так что полный разброс есть сумма разброса внутри класса и разброса между классами:

$$S_B = \sum_{i=1}^{M} n_i (\bar{m}_i - \bar{m})(\bar{m}_i - \bar{m})^t$$

и

$$S_T = S_W + S_B.$$

Проекция из *d*-мерного пространства в *(M—1)*-мерное пространство осуществляется с помощью М—1 разделяющих функций

$$y_i = \bar{w}_i^t \bar{x}; \qquad i = 1, \ldots\ldots M - 1.$$

Если считать $y_i$ составляющими вектора $\bar{y}$, а векторы весовых функций $\bar{w}_i$ столбцами матрицы $W$ размера $d \times (M-1)$, то проекцию можно записать в виде одного матричного уравнения

$$\bar{y} = W^t \bar{x}.$$

Выборки $y_1, \ldots, y_n$ проецируются на соответствующее множество выборок $x_1, \ldots\ldots x_n$, которые можно описать с помощью их векторов средних значений и матриц разброса. Так, если мы определяем

$$\tilde{m}_i = \frac{1}{n_i}\sum_{y\in Y_i} y;$$

$$\tilde{m} = \frac{1}{n}\sum_{i=1}^{M} n_i \tilde{m}_i ;$$

$$\tilde{S}_W = \sum_{i=1}^{M} \sum_{y \in Y_i} (y - \tilde{m}_i)(y - \tilde{m}_i)^t ;$$

и

$$\tilde{S}_B = \sum_{i=1}^{M} n_i (\tilde{m}_i - \tilde{m})(\tilde{m}_i - \tilde{m})^t ,$$

то можно непосредственно получить

$$\tilde{S}_W = W^t S_W W ;$$

$$\tilde{S}_B = W^t S_B W .$$

Эти уравнения показывают, как матрицы разброса внутри класса и между классами отображаются посредством проекции в пространство меньшей размерности. Находим матрицу отображения $W$, которая в некотором смысле максимизирует отношение разброса между классами к разбросу внутри класса. Простым скалярным показателем разброса является определитель матрицы разброса. Определитель есть произведение собственных значений, а следовательно, и произведение дисперсий в основных направлениях, измеряющее объем гиперэллипсоида разброса. Пользуясь этим показателем, получим функцию критерия

$$J(W) = \frac{\left|\tilde{S}_B\right|}{\left|\tilde{S}_W\right|} = \frac{\left|W^t S_B W\right|}{\left|W^t S_W W\right|} .$$

Задача нахождения прямоугольной матрицы $W$, которая максимизирует $J$, достаточно сложна. К счастью, оказывается, что ее решение имеет относительно простой вид. Столбцы оптимальной матрицы $W$ являются обобщенными собственными векторами, соответствующими наибольшим собственным значениям в

$$S_B w_i = \lambda_i S_W w_i .$$

Следует сделать несколько замечаний относительно данного решения. Во-первых, если $S_w$ – невырожденная матрица, то данную задачу можно свести к обычной задаче определения собственного значения. Однако в

действительности это нежелательно, т.к. при этом потребуется ненужное вычисление матрицы, обратной $S_w$. Вместо этого можно найти собственные значения как корни характеристического полинома

$$\left| S_B - \lambda_i S_W \right| = 0;$$

а затем решить

$$(S_B - \lambda_i S_W) w_i = 0;$$

непосредственно для собственных векторов $w_i$. Поскольку $S_B$ является суммой M матриц ранга единица или менее и поскольку только M–1 из них независимые матрицы, постольку $S_B$ имеет ранг (M–1) или меньше. Так что не более M–1 собственных значений не нули и искомые векторы весовых функций соответствуют этим ненулевым собственным значениям. Если разброс внутри класса изотропный, то собственные векторы будут просто собственными векторами матрицы $S_B$, а собственные векторы с ненулевыми собственными значениями стягивают пространство, натянутое на векторы $m_i$—$m$. В этом частном случае столбцы матрицы $W$ можно найти, просто применяя процедуру ортонормирования Грама–Шмидта к M–1 векторам $m_i$. Наконец, заметим, что, вообще говоря, решение для W не является однозначным. Допустимые преобразования включают вращение и масштабирование осей различными путями. Это все линейные преобразования из (M—1)-мерного пространства в (M–1)-мерное пространство, и они не меняют значительно положения вещей. В частности, они оставляют функцию критерия $J(W)$ инвариантной.

Как и в случае с двумя классами, множественный дискриминантный анализ в первую очередь позволяет сократить размерность задачи. Параметрические или непараметрические методы, которые могут не сработать в первоначальном (многомерном) пространстве, могут хорошо действовать в пространстве меньшей размерности.

Например можно будет оценить отдельные ковариационные матрицы для каждого класса и использовать допущение об общем многомерном нормальном распределении после преобразования, что было невозможно сделать с первоначальными данными. Вообще преобразование влечет за собой некоторое

ненужное перемешивание данных и повышает теоретически достижимый уровень ошибки, а проблема классификации данных все еще остается.

# 5. Нейронные сети

## *5.1. Общие принципы построения нейронной сети*

Одним из мощных методов принятия решения классификации наблюдаемых объектов являются нейронные сети, основанные на идее моделирования процессов, происходящих в мозгу человека, поскольку тот факт, что возможности человеческого мозга распознавать объекты находятся вне конкуренции по сравнению с любым сколь угодно мощным вычислительным средством, не вызывает сомнения.

Нервная система и мозг человека состоят из нейронов, соединенных между собой нервными волокнами. Нервные волокна способны передавать электрические импульсы между нейронами. Все процессы передачи раздражений от нашей кожи, ушей и глаз к мозгу, процессы мышления и управления действиями – все это реализовано в живом организме как передача электрических импульсов между нейронами. Рассмотрим строение биологического нейрона.

Каждый нейрон имеет отростки нервных волокон двух типов: дендриты, по которым принимаются импульсы, и единственный аксон, по которому нейрон может передавать импульс. Аксон контактирует с дендритами других нейронов через специальные образования – синапсы, которые влияют на силу импульса.

Можно считать, что при прохождении синапса сила импульса меняется в определенное число раз, которое мы будем называть весом синапса. Импульсы, поступившие к нейрону одновременно по нескольким дендритам, суммируются. Если суммарный импульс превышает некоторый порог, нейрон возбуждается, формирует собственный импульс и передает его далее по аксону.

Важно отметить, что веса синапсов может изменяться со временем, а значит меняется и поведение соответствующего нейрона.

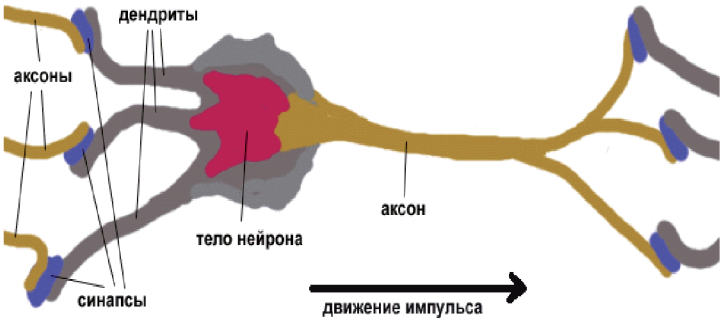



Рис. 5.1

Нетрудно построить математическую модель описанного процесса.

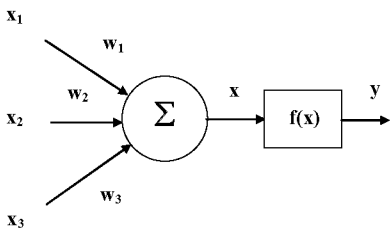



Рис. 5.2

На рисунке 5.2 изображена модель нейрона с тремя входами (дендритами); синапсы этих дендритов имеют веса $w_1$, $w_2$, $w_3$. Пусть к синапсам поступают импульсы силы $x_1$, $x_2$, $x_3$ соответственно, тогда после прохождения синапсов и дендритов к нейрону поступают импульсы $w_1x_1$, $w_2x_2$, $w_3x_3$. Нейрон преобразует полученный суммарный импульс $x=w_1x_1+ w_2x_2+ w_3x_3$ в соответствии с некоторой передаточной функцией $f(x)$. Сила выходного импульса равна $y = f(x) = f(w_1x_1 + w_2x_2 + w_3x_3)$.

Таким образом, нейрон полностью описывается своими весами $w_k$ и передаточной функцией $f(x)$. Получив набор чисел (вектор) $x_k$ в качестве входов, нейрон выдает некоторое число y на выходе.

Искусственная нейронная сеть (ИНС, нейронная сеть) – это набор нейронов, соединенных между собой. Как правило, передаточные функции всех нейронов в нейронной сети фиксированы, а веса являются параметрами нейронной сети и могут изменяться. Некоторые входы нейронов помечены как внешние входы нейронной сети, а некоторые выходы – как внешние выходы нейронной сети. Подавая любые числа на входы нейронной сети, мы получаем какой-то набор чисел на выходах нейронной сети. Таким образом, работа нейронной сети состоит в преобразовании входного вектора в выходной вектор, причем это преобразование задается весами нейронной сети.

Практически любую задачу можно свести к задаче, решаемой с помощью нейронной сети. В качестве примера сформулируем в терминах нейронной сети задачу распознавания рукописных букв.

Дано: растровое черно-белое изображение буквы размером 30 x 30 пикселов (входной вектор из 900 двоичных символов 30 x 30 = 900). Требуется определить, какая это буква (в алфавите 33 буквы). Задача заключается в том, чтобы построить нейронную сеть с 900 входами и 33 выходами, которые помечены буквами. Если на входе нейронной сети присутствует изображение буквы "А", то максимальное значение выходного сигнала достигается на выходе "А". Так же нейронная сеть работает для всех 33 букв.

Поясним, зачем требуется выбирать выход нейронной сети с максимальным уровнем сигнала. Дело в том, что уровень выходного сигнала, как правило, может принимать любые значения из какого-то отрезка. Однако в данной задаче нас интересует не аналоговый ответ, а всего лишь номер категории (номер буквы в алфавите). Поэтому используется следующий подход: каждой категории сопоставляется свой выход, а ответом нейронной сети считается та категория, на чьем выходе уровень сигнала максимален. В

определенном смысле уровень сигнала на выходе "А" – это достоверность того, что на вход нейронной сети была подана рукописная буква "А".

Теперь, когда стало ясно, что именно мы хотим построить, можем переходить к вопросу, как строить такую нейронную сеть. Этот вопрос решается в два этапа:

1) выбор типа (архитектуры) нейронной сети;
2) подбор весов (обучение) нейронной сети.

На первом этапе следует выбрать следующее:

- какие нейроны мы хотим использовать (число входов, передаточные функции);
- каким образом следует соединить их между собой;
- что взять в качестве входов и выходов нейронной сети.

Для решения данных задач – существует несколько десятков различных нейросетевых архитектур, причем эффективность многих из них доказана математически. Наиболее популярные и изученные архитектуры – это многослойный перцептрон, нейронная сеть с общей регрессией, нейронная сеть Кохонена и другие.

На втором этапе нам следует "обучить" выбранную нейронную сеть, т. е. подобрать такие значения ее весов, чтобы она работала нужным образом. В используемых на практике нейронных сетях количество весов может составлять несколько десятков тысяч, поэтому обучение – действительно сложный процесс. Для многих архитектур разработаны специальные алгоритмы обучения, которые позволяют настроить веса нейронной сети определенным образом. Наиболее популярный из таких алгоритмов метод обратного распространения ошибки (Error Back Propagation), используемый, например, для обучения перцептрона.

Обучение нейронной сети заключается в том, чтобы сообщить ей, чего мы от нее добиваемся. Этот процесс очень похож на обучение ребенка алфавиту. Показав ребенку изображение буквы А, мы спрашиваем его: "Какая это буква?" Если ответ неверен, мы сообщаем ребенку тот ответ, который

хотели бы от него получить: "Это буква А". Ребенок запоминает пример вместе с верным ответом, т. е. в его памяти происходят некоторые изменения в нужном направлении. Мы будем повторять процесс предъявления букв снова и снова до тех пор, пока все 33 буквы не будут твердо запомнены. Такой процесс называют "обучение с учителем".

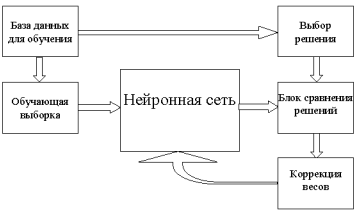



Рис. 5.3

При обучении нейронной сети мы действуем совершенно аналогично. Имеется некоторая база данных, содержащая примеры (набор рукописных изображений букв). Предъявляя изображение буквы "А" на вход нейронной сети, мы получаем от нее некоторый ответ, не обязательно верный. Нам известен и верный (желаемый) ответ – в данном случае нужно, чтобы на выходе нейронной сети с меткой "А" уровень сигнала был максимален. Обычно в качестве желаемого выхода в задаче классификации берут набор (1, 0, 0, ...), где 1 стоит на выходе с меткой "А", а 0 – на всех остальных выходах. Вычисляя разность между желаемым ответом и реальным ответом сети, мы получаем 33 числа – *вектор ошибки*. Алгоритм обратного распространения ошибки – это набор формул, который позволяет по вектору ошибки вычислить требуемые поправки для весов нейронной сети. Одну и ту же букву (а также различные изображения одной и той же буквы) мы можем предъявлять нейронной сети много раз. В этом смысле обучение скорее напоминает повторение упражнений в спорте – тренировку. Блок-схема процесса обучения нейронной сети приведена на рис. 5.3.

После многократного предъявления примеров веса нейронной сети стабилизируется, причем нейронная сеть дает правильные ответы на все (или

почти все) примеры из базы данных. В таком случае говорят, что "нейронная сеть выучила все примеры", "нейронная сеть обучена" или "нейронная сеть натренирована". В программных реализациях можно видеть, что в процессе обучения величина ошибки (сумма квадратов ошибок по всем выходам) постепенно уменьшается. Когда величина ошибки достигает нуля или приемлемого малого уровня, тогда тренировку останавливают, а полученную нейронную сеть считают натренированной и готовой к применению на новых данных.

Важно отметить, что вся информация, которую нейронная сеть имеет о задаче, содержится в наборе примеров. Поэтому качество обучения нейронной сети напрямую зависит от количества примеров в обучающей выборке, а также от того, насколько полно эти примеры описывают данную задачу. Так, например, бессмысленно использовать нейронную сеть для предсказания финансового кризиса, если в обучающей выборке кризисов не представлено. Считается, что для полноценной тренировки нейронной сети требуется хотя бы несколько десятков (а лучше сотен) примеров.

Повторим еще раз, что обучение нейронных сетей – сложный и наукоемкий процесс. Алгоритмы обучения нейронных сетей имеют различные параметры и настройки, для управления которыми требуется понимание их влияния.

После того как нейронная сеть обучена, мы можем применять ее для решения полезных задач. Важнейшая особенность человеческого мозга состоит в том, что, однажды обучившись определенному процессу, он может верно действовать и в тех ситуациях, в которых он не бывал в процессе обучения. Например, мы можем читать почти любой почерк, даже если видим его первый раз в жизни. Так же и нейронная сеть, грамотным образом обученная, может с большой вероятностью правильно реагировать на новые, не предъявленные ей ранее данные. Например, мы можем нарисовать букву "А" другим почерком, а затем предложить нашей нейронной сети классифицировать новое изображение. Веса обученной нейронной сети хранят достаточно много

информации о сходстве и различиях букв, поэтому мы можем рассчитывать на правильный ответ и для нового варианта изображения.

Описанные выше процессы обучения и применения нейронных сетей можно увидеть в действии прямо сейчас Для этого достаточно кликнуть по приведенной ссылке: http://www.neuroproject.ru/download.htm#dnwsgdemo и проанализировать несколько простых программ, подготовленных фирмой Ward Systems Group на основе библиотеки NeuroWindows. Каждая из программ позволяет пользователю самостоятельно задать набор примеров и обучить на этом наборе определенную нейронную сеть. Затем можно ей предлагать новые примеры и наблюдать работу нейронной сети.

## *5.2. Области применения нейронных сетей*

### 5.2.1. Нейросетевая классификация

Отметим, что задачи классификации (например, распознавания букв) плохо формализуются. Если в случае распознавания букв верный ответ очевиден заранее, то в более сложных практических задачах обученная нейронная сеть выступает как эксперт, обладающий большим опытом и способный дать ответ на трудный вопрос.

Примером такой задачи служит медицинская диагностика, где нейронная сеть может учитывать большое количество числовых параметров (энцефалограмма, давление, вес и т. д.). Конечно, "мнение" нейронной сети в этом случае нельзя считать окончательным.

Классификация предприятий по степени их перспективности – это уже привычный способ использования нейронных сетей в практике западных компаний. В таком случае нейронная сеть также использует множество экономических показателей, сложным образом связанных между собой.

Нейросетевой подход особенно эффективен в задачах экспертной оценки по той причине, что он сочетает в себе способность компьютера к обработке чисел и способность мозга к обобщению и распознаванию. Говорят, что у хорошего врача способность к распознаванию в своей области столь велика,

что он может провести приблизительную диагностику уже по внешнему виду пациента. Можно согласиться также, что опытный трейдер чувствует направление движения рынка по виду графика. Однако в первом случае все факторы наглядны, т. е. характеристики пациента мгновенно воспринимаются мозгом бледное лицо, блеск в глазах и т. д. В другом же случае учитывается только один фактор – курс за определенный период времени. Нейронная сеть позволяет обрабатывать огромное количество факторов (до нескольких тысяч), независимо от их наглядности – это универсальный "хороший врач", который может поставить свой диагноз в любой области.

### 5.2.2. Кластеризация с помощью нейронных сетей и поиск зависимостей

Помимо задач классификации, нейронные сети широко используются для поиска зависимостей в данных и кластеризации. Например, нейронная сеть на основе методики МГУА (метод группового учета аргументов) позволяет на основе обучающей выборки построить зависимость одного параметра от других в виде полинома.

Такая нейронная сеть может не только мгновенно выучить таблицу умножения, но и найти сложные скрытые зависимости в данных (например, финансовых), которые не обнаруживаются стандартными статистическими методами.

Кластеризация – это разбиение набора примеров на несколько компактных областей (кластеров), причем число кластеров заранее неизвестно.

Кластеризация позволяет представить неоднородные данные в более наглядном виде и использовать далее для исследования каждого кластера различные методы. Например, таким образом можно быстро выявить фальсифицированные страховые случаи или недобросовестные предприятия.

### 5.2.3. Применение нейронных сетей в задачах прогнозирования

Задачи прогнозирования особенно важны для практики, в частности, для финансовых приложений, поэтому рассмотрим способы применения нейронных сетей в этой области более подробно.

Дана задача прогнозирования курса акций на 1 день вперед.

Пусть у нас имеется база данных, содержащая значения курса за последние 300 дней. Простейший вариант в данном случае – попытаться построить прогноз завтрашней цены на основе курсов за последние несколько дней. Понятно, что прогнозирующая нейронная сеть должна иметь всего один выход и столько входов, сколько предыдущих значений мы хотим использовать для прогноза, например 4 последних значения. Составить обучающий пример очень просто – входными значениями нейронной сети будут курсы за 4 последовательных дня, а желаемым выходом нейронной сети – известный нам курс в следующий день за этими четырьмя.

Если нейронная сеть совместима с какой-либо системой обработки электронных таблиц (например, Excel), то подготовка обучающей выборки состоит из следующих операций:

- скопировать столбец данных значений котировок в 4 соседних столбца;
- сдвинуть второй столбец на 1 ячейку вверх, третий столбец – на 2 ячейки вверх и т. д.

Пример этой подготовки легко увидеть на рисунке (рис. 5.4) – теперь каждая строка таблицы представляет собой обучающий пример, где первые 4 числа – входные значения нейронные сети, а пятое число – желаемое значение выхода нейронной сети. Исключение составляют последние 4 строки, где данных недостаточно – эти строки не учитываются при тренировке нейронной сети. Отметим, что в четвертой снизу строке заданы все 4 входных значения, но неизвестно значение выхода нейронной сети. Именно к ней мы применим обученную нейронную сеть и получим прогноз на следующий день.

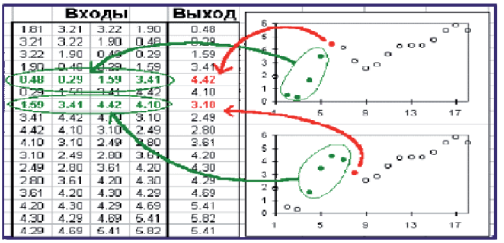



Рис. 5.4

Как видно из этого примера, объем обучающей выборки зависит от выбранного нами количества входов нейронной сети. Если сделать 299 входов, то такая нейронная сеть потенциально могла бы строить лучший прогноз, чем нейронная сеть с 4 входами, однако в таком случае мы имеем всего 1 обучающий пример и обучение бессмысленно. При выборе числа входов нейронной сети это следует учитывать, выбирая разумный компромисс между глубиной предсказания (число входов нейронной сети) и качеством обучения нейронной сети (объем тренировочного набора).

### *5.3. Алгоритм обратного распространения ошибки*

Названный алгоритм является, пожалуй, наиболее широко распространенным алгоритмом обучения многослойных нейронных сетей. Формулу, описывающую функционирование нейрона, запишем в виде

$$y = \begin{cases} 1 \text{ при } \sum_{i=1}^{N} w_i u_i \geq v; \\ 0 \text{ при } \sum_{i=1}^{N} w_i u_i \leq v. \end{cases} \qquad (5.1)$$

Модель (5.1) может быть представлена в виде

$$y = f\left(\sum_{i=0}^{N} w_i u_i\right), \qquad (5.2)$$

где

$$f(x) = \begin{cases} 1 \text{ при } x \geq 0 \\ 0 \text{ при } x < 0 \end{cases}.$$

Формула (5.2) описывает модель нейрона, представленную на рис. 5.5. В качестве функции *f(x)*, как правило, используется сигмоидальная функция, определяемая выражением

$$f(x) = \frac{1}{1+e^{-\beta x}}.$$

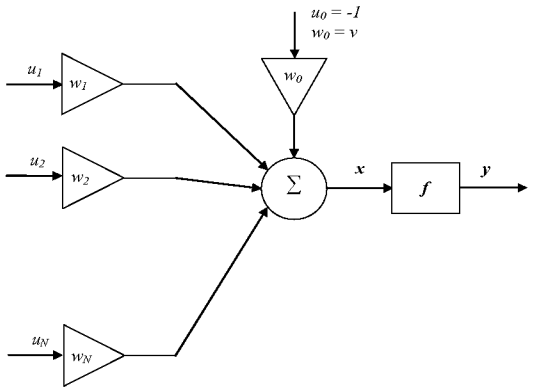



Рис. 5.5

На рис. 5.6 представлена многослойная нейронная сеть, состоящая из $L$ слоев. В каждом слое расположено $N_k$ элементов, $k = 1, 2, \ldots, L$, обозначаемых $AD_i^k$, $i = 1, 2, \ldots, N_k$. Каждый из таких элементов представляет собой нейрон, структура которого аналогична нейрону, изображенному на рис. 5.5.

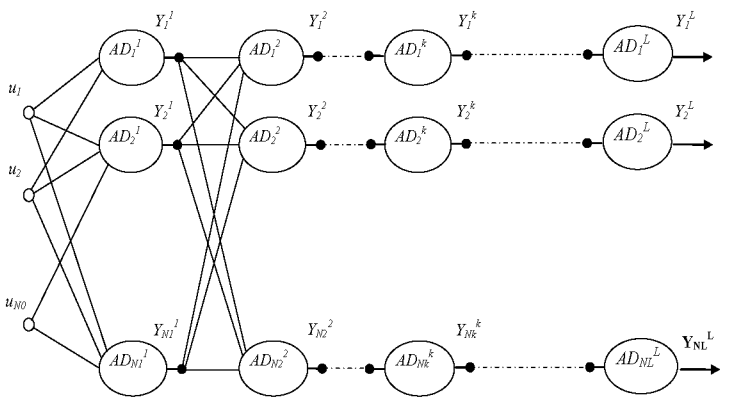



Рис. 5.6

Рассматриваемая нейронная сеть имеет $N_0$ входов, на которые подаются сигналы $u_1(n)$, …, $u_{N0}(n)$, записываемые в векторной форме

$$U = \left[u_1(n),.....u_{N_0}(n)\right]^T, \quad n = 1,2,....$$

Выходной сигнал $i$-го нейрона в $k$-м слое обозначается $y_i^{(k)}(n)$, $i = 1,\ldots, N_k$, $k = 1,\ldots, L$.

Нейрон $AD_i^k$ имеет $N_k$ входов, образующих вектор

$$X^{(k)}(n) = \left[x_0^{(k)}(n),.....,x_{N_{k-1}}^{(k)}(n)\right]^T,$$

причем $x_j^{(k)}(n) = +1$ для $j = 0$ и $k = 1, \ldots, L$.

Входной сигнал нейрона $\mathbf{AD_i^k}$ связан с выходным сигналом (k-1) слоя следующим образом:

$$x_j^k(n) = \begin{cases} u_i(n) & для \quad k = 1, \\ y_j^{(k-1)} & для \quad k = 2,.....L, \\ +1 & для \quad j = 0, \ k = 1,.....L \end{cases} .$$

Обозначим $w_{ij}^{(k)}(n)$ вес связи $i$-го нейрона, $i = 1,\ldots\ldots N_k$, расположенного в $k$-м слое, с $j$-м входным сигналом $x_j^{(k)}(n)$, $j = 0, 1,\ldots, N_{k1}$. Вектор весов нейрона $AD_i^k$ обозначим

$$W_i^k(n) = \left[w_{i,0}^{(k)}(n),.....w_{i,N_{k-1}}^{(k)}\right]^T, \ k = 1,.....L, i = 1,.....N_k .$$

Выходной сигнал нейрона $AD_i^k$ в n-й момент времени, n=1,2… определяется как

$$y_i^{(k)}(n) = f\left(s_i^{(k)}(n)\right), \tag{5.3}$$

причем

$$s_i^{(k)}(n) = \sum_{j=0}^{N_{k-1}} w_{ij}^{(k)}(n) x_j^{(k)}(n) . \tag{5.4}$$

Выходные сигналы нейронов в $L$-м слое

$$y_1^L(n), y_2^L(n),.....y_{N_L}^L(n) \tag{5.5}$$

одновременно являются выходными сигналами всей сети. Они сравниваются с так называемыми эталонными сигналами сети

$$d_1^L(n), d_2^L(n),.....d_{N_L}^L(n) \tag{5.6}$$

в результате чего получаем погрешность

$$\varepsilon_i^L(n)=d_i^L(n)-y_i^L(n),....i=1,.....N_L. \qquad (5.7)$$

Сформулируем меру погрешности, основанную на сравнении сигналов (5.5) и (5.6), в виде суммы квадратов разностей (5.7):

$$Q(n)=\sum_{i=1}^{N_L}\left(\varepsilon_i^{(L)}\right)^2(n)=\sum_{i=1}^{N_L}\left(d_i^L(n)-y_i^L(n)\right)^2. \qquad (5.8)$$

Из выражений (5.3), (5.4) следует, что мера погрешности (5.8) является функцией весов сети. Обучение сети основано на адаптивной коррекции всех весов $w_{ij}^{(k)}(n)$ таким образом, чтобы минимизировать меру погрешности. Для коррекции произвольного веса можно использовать правило наискорейшего спуска в следующем виде:

$$w_{ij}^{(k)}(n+1)=w_{ij}^{(k)}(n)-\eta\frac{\partial Q(n)}{\partial w_{ij}^{(k)}(n)}, \qquad (5.9)$$

где константа $\eta > 0$ определяет величину шага коррекции.

Вычислим производную в (5.9):

$$\frac{\partial Q(n)}{\partial w_{ij}^{(k)}(n)}=\frac{\partial Q(n)}{\partial s_i^{(k)}(n)}\frac{\partial s_i^{(k)}(n)}{\partial w_{ij}^{(k)}(n)}=\frac{\partial Q(n)}{\partial s_i^{(k)}(n)}x_j^{(k)}(n).$$

Если ввести обозначение

$$\delta_i^{(k)}(n)=-\frac{1}{2}\frac{\partial Q(n)}{\partial s_i^{(k)}(n)}, \qquad (5.10)$$

то получим равенство

$$\frac{\partial Q(n)}{\partial w_{ij}^{(k)}(n)}=-2\delta_i^{(k)}(n)x_j^{(k)}(n).$$

При этом алгоритм (5.9) принимает вид

$$w_{ij}^{(k)}(n+1)=w_{ij}^{(k)}(n)+2\eta\delta_i^{(k)}(n)x_j^{(k)}(n).$$

Способ расчета значения $\delta_i^{(k)}(n)$, заданного выражением (5.10), зависит от номера слоя. Для последнего слоя получаем

$$\delta_i^L(n) = -\frac{1}{2}\frac{\partial Q(n)}{\partial s_i^L(n)} = -\frac{1}{2}\frac{\partial \sum_{m=1}^{N_L}\left(\varepsilon_m^L\right)^2(n)}{\partial s_i^L(n)} =$$

$$= -\frac{1}{2}\frac{\partial\left(\varepsilon_i^L\right)^2(n)}{\partial s_i^L(n)} = -\frac{1}{2}\frac{\partial\left(d_i^L(n) - y_i^L(n)\right)^2}{\partial s_i^L(n)} =$$

$$= \varepsilon_i^L(n)\frac{\partial y_i^L(n)}{\partial s_i^L(n)} = \varepsilon_i^L(n) f'\left(s_i^L(n)\right).$$

Для произвольного слоя $k \neq L$ получаем

$$\delta_i^{(k)}(n) = -\frac{1}{2}\frac{\partial Q(n)}{\partial s_i^{(k)}(n)} = -\frac{1}{2}\sum_{m=1}^{N_{k+1}}\frac{\partial Q(n)}{\partial s_m^{(k+1)}(n)}\frac{\partial s_m^{(k+1)}(n)}{\partial s_j^{(k)}(n)} =$$

$$= \sum_{m=1}^{N_{k+1}}\delta_m^{(k+1)}(n) w_{mi}^{k+1}(n) f'\left(s_i^{(k)}(n)\right) = \qquad (5.11)$$

$$= f'\left(s_i^{(k)}(n)\right)\sum_{m=1}^{N_{k+1}}\delta_m^{(k+1)}(n) w_{mi}^{k+1}(n).$$

Определим погрешность в k-м (не последнем) слое для i-го нейрона в виде

$$\varepsilon_i^{(k)}(n) = \sum_{m=1}^{N_{k+1}}\delta_m^{(k+1)}(n) w_{mi}^{k+1}(n), \quad k = 1,\ldots,(L-1). \qquad (5.12)$$

Если подставить выражение (5.12) в (5.11), то получим

$$\delta_i^{(k)}(n) = \varepsilon_i^{(k)}(n) f'\left(s_i^{(k)}(n)\right).$$

В результате алгоритм обратного распространения ошибки можно записать в виде

$$y_i^{(k)}(n) = f\left(s_i^{(k)}(n)\right), \qquad s_i^{(k)}(n) = \sum_{j=0}^{N_{k-1}} w_{ij}^{(k)}(n) x_j^{(k)}(n)$$

$$\varepsilon_i^{(k)}(n) = \begin{cases} d_i^L(n) - y_i^L(n) & для \;\; k = L \\ \sum_{m=1}^{N_{k+1}}\delta_m^{k+1}(n) w_{mi}^{k+1}(n) & для \;\; k = 1,2,\ldots,L \end{cases}$$

$$\delta_i^k(n) = \varepsilon_i^{(k)}(n) f'\left(s_i^{(k)}(n)\right)$$

$$w_{ij}^{(k)}(n+1) = w_{ij}^{(k)}(n) + 2\eta\delta_i^{(k)}(n) x_j^{(k)}(n).$$

Идея алгоритма связана со способом расчета погрешностей в конкретных слоях. В первую очередь рассчитываются погрешности в последнем слое (на основе выходных и эталонных сигналов), далее – в предпоследнем и так до первого слоя. Начальные значения весов выбираются случайными и, как правило, устанавливаются близкими к нулю.

# 6. Генетические алгоритмы

## *6.1. Природа генетических алгоритмов*

Нейронные сети были созданы в результате наблюдения за естественными процессами, происходящими в нервной системе живых существ, и попыток воспроизведения этих процессов. Термин “нейрон”, обозначающий основной исполнительный элемент искусственных нейронных сетей, был непосредственно заимствован из теории природных нервных систем.

Аналогично генетические алгоритмы возникли в результате наблюдения и попыток копирования естественных процессов, происходящих в мире живых организмов, в частности, эволюции и связанной с ней селекции (естественного отбора) популяций живых существ. Конечно, при подобном сопоставлении нейронных сетей и генетических алгоритмов следует обращать внимание на принципиально различную длительность протекания упоминаемых естественных процессов, т. е. на чрезвычайно быструю обработку информации в нервной системе и очень медленный процесс естественной эволюции. Однако при компьютерном моделировании эти различия оказываются несущественными.

Идею генетических алгоритмов высказал Дж. Холланд в кон. 60-х – нач. 70-х гг. XX в. Он заинтересовался свойствами процессов естественной эволюции (в т.ч. фактом эволюции хромосом, а не самих живых существ). Холланд был уверен в возможности составить и реализовать в виде компьютерной программы алгоритм, который будет решать сложные задачи так, как это делает природа - путем эволюции. Поэтому он начал трудиться над алгоритмами, оперировавшими последовательностями двоичных цифр (единицы и нули), получившими название хромосом. Эти алгоритмы имитировали эволюционные процессы в поколениях таких хромосом. В них были реализованы механизмы селекции и репродукции, аналогичные применяемым при естественной эволюции. Так же, как и в природе, генетические алгоритмы

осуществляли поиск “хороших” хромосом без использования какой-либо информации о характере решаемой задачи. Требовалась только некая оценка каждой хромосомы, отражающая ее приспособленность. Механизм селекции заключается в выборе хромосом с наивысшей оценкой (т. е. наиболее приспособленных), которые репродуцируют чаще, чем особи с более низкой оценкой (хуже приспособленные). Репродукция означает создание новых хромосом в результате рекомбинации генов родительских хромосом.

*Рекомбинация* – это процесс, в результате которого возникают новые комбинации генов. Для этого используются две операции: *скрещивание,* позволяющее создать две совершенно новые хромосомы потомков путем комбинирования генетического материала пары родителей, а также *мутация,* которая может вызывать изменения в отдельных хромосомах.

В генетических алгоритмах применяется ряд терминов, заимствованных из генетики: прежде всего *гены* и *хромосомы,* а также *популяция, особь, аллель, генотип, фенотип.*

Генетические алгоритмы применяются при разработке программного обеспечения, в системах искусственного интеллекта, оптимизации, искусственных нейронных сетях и в других отраслях знаний. Следует отметить, что с их помощью решаются задачи, для которых ранее использовались только нейронные сети. В этом случае генетические алгоритмы выступают просто в роли независимого от нейронных сетей альтернативного метода, предназначенного для решения той же самой задачи. Генетические алгоритмы часто используются совместно с нейронными сетями. Они могут поддерживать нейронные сети, или оба метода могут взаимодействовать в рамках гибридной системы, предназначенной для решения конкретной задачи.

Эволюционная теория утверждает, что каждый биологический вид целенаправленно развивается и изменяется для того, чтобы наилучшим образом приспособиться к окружающей среде. В процессе эволюции многие виды насекомых и рыб приобрели защитную окраску, еж стал неуязвимым благодаря иглам, человек стал обладателем сложнейшей нервной системы. Можно

сказать, что эволюция – это процесс оптимизации всех живых организмов. Рассмотрим, какими же средствами природа решает эту задачу оптимизации.

Основной механизм эволюции – естественный отбор. Его суть состоит в том, что более приспособленные особи имеют больше возможностей для выживания и размножения и, следовательно, приносят больше потомства, чем плохо приспособленные особи. При этом благодаря передаче генетической информации (*генетическое наследование*) потомки наследуют от родителей основные их качества. Таким образом, потомки сильных индивидуумов также будут относительно хорошо приспособленными, а их доля в общей массе особей будет возрастать. После смены нескольких десятков или сотен поколений средняя приспособленность особей данного вида заметно возрастает.

Чтобы сделать понятными принципы работы генетических алгоритмов, поясним также, как устроены механизмы генетического наследования в природе. В каждой клетке любого животного содержится вся генетическая информация особи. Такая информация записана в виде набора очень длинных молекул ДНК (дезоксирибонуклеиновая кислота). Каждая молекула ДНК – это цепочка, состоящая из молекул *нуклеотидов* четырех типов, обозначаемых *A, T, C* и *G*. Собственно информацию несет порядок следования нуклеотидов в ДНК. Таким образом, генетический код индивидуума – это просто очень длинная строка символов, в которой используется всего 4 буквы. В животной клетке каждая молекула ДНК окружена оболочкой – такое образование называется *хромосомой.*

Любое врожденное качество особи (цвет глаз, наследственные болезни, тип волос и т. д.) кодируется определенной частью хромосомы, которая называется *геном* этого свойства. Например, ген цвета глаз содержит информацию, кодирующую определенный цвет глаз. Различные значения гена называются его *аллелями*.

При размножении животных происходит слияние двух родительских половых клеток и их ДНК взаимодействуют, образуя ДНК потомка. Основной

способ взаимодействия – *кроссовер (cross-over, скрещивание).* При кроссовере ДНК предков делятся на две части, а затем обмениваются своими половинками.

При наследовании возможны мутации из-за радиоактивности и других воздействий, в результате которых могут измениться некоторые гены в половых клетках одного из родителей. Измененные гены передаются потомку и придают ему новые свойства. Если эти новые свойства полезны, они, скорее всего, сохранятся в данном виде. При этом произойдет скачкообразное повышение приспособленности вида.

Пусть дана некоторая сложная функция (*целевая функция*), зависящая от нескольких переменных, и требуется найти такие значения переменных, при которых значение функции максимально. Задачи такого рода называются *задачами оптимизации* и встречаются на практике очень часто.

Чтобы смоделировать эволюционный процесс, сгенерируем вначале случайную популяцию – несколько индивидуумов со случайным набором хромосом (числовых векторов). Генетический алгоритм имитирует эволюцию этой популяции как циклический процесс скрещивания индивидуумов и смены поколений (рис. 6.1).

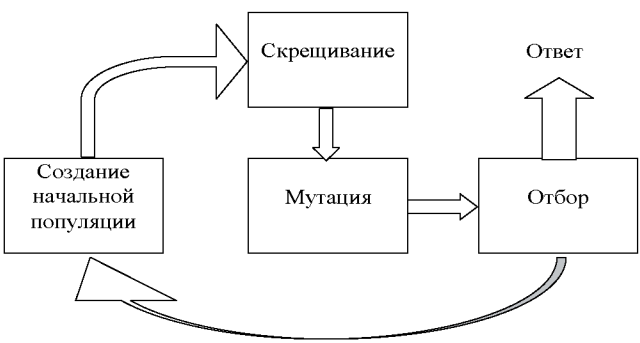



Рис. 6.1

*Жизненный цикл популяции* – это несколько случайных скрещиваний (посредством кроссовера) и мутаций, в результате которых к популяции добавляется какое-то количество новых индивидуумов. Отбор в генетическом алгоритме – процесс формирования новой популяции из старой, после чего старая популяция погибает. После отбора к новой популяции опять применяются операции кроссовера и мутации, затем опять происходит отбор, и так далее.

Отбор в генетическом алгоритме тесно связан с принципами естественного отбора в природе следующим образом:

- приспособленность индивидуума – значение целевой функции на этом индивидууме.
- выживание наиболее приспособленных – популяция следующего поколения формируется в соответствии с целевой функцией. Чем приспособленнее индивидуум, тем больше вероятность его участия в кроссовере.

Таким образом, модель отбора определяет, каким образом следует строить популяцию следующего поколения. Как правило, вероятность участия индивидуума в скрещивании берется пропорционально его приспособленности. Часто используется так называемая *стратегия элитизма*, при которой несколько лучших индивидуумов переходят в следующее поколение без изменений, не участвуя в кроссовере и отборе. В любом случае каждое следующее поколение будет в среднем лучше предыдущего. Когда приспособленность индивидуумов перестает заметно увеличиваться, тогда процесс останавливают и в качестве решения задачи оптимизации берут наилучшего из найденных индивидуумов.

## *6.2. Генетические алгоритмы и традиционные методы оптимизации*

Генетический алгоритм представляет собой метод, отражающий естественную эволюцию методов решения проблем и, в первую очередь, задач

оптимизации. *Генетические алгоритмы* – это процедуры поиска, основанные на механизмах естественного отбора и наследования. В них используется эволюционный принцип выживания наиболее приспособленных особей. Они отличаются от традиционных методов оптимизации несколькими базовыми элементами. В частности, генетические алгоритмы:

- обрабатывают не значения параметров самой задачи, а их закодированную форму;
- осуществляют поиск решения исходя не из единственной точки, а из их некоторой популяции;
- используют только целевую функцию, а не ее производные либо иную дополнительную информацию;
- применяют вероятностные, а не детерминированные правила выбора.

Перечисленные четыре свойства, которые можно сформулировать так же, как кодирование параметров, операции на популяциях, использование минимума информации о задаче и рандомизации операций, приводят к устойчивости генетических алгоритмов и к их превосходству над другими широко применяемыми технологиями.

### *6.3. Основные понятия генетических алгоритмов*

При описании генетических алгоритмов используются определения, заимствованные из генетики. Например, речь идет о популяции особей, а в качестве базовых понятий применяются ген, хромосома, генотип, фенотип, аллель. Также используются соответствующие этим терминам определения из технического лексикона, в частности, цепь, двоичная последовательность, структура.

*Популяция* – это конечное множество *особей.*

*Особи,* входящие в популяцию, в генетических алгоритмах представляются хромосомами с закодированными в них множествами *параметров задачи,* т. е. решений, которые иначе называются *точками в пространстве поиска (search points).* В некоторых работах особи называются *организмами.*

*Хромосомы* (*цепочки,* или *кодовые последовательности)* – это упорядоченные *последовательности генов.*

*Ген* (также называемый *свойством, знаком* или *детектором)* – это атомарный элемент *генотипа,* в частности, хромосомы.

*Генотип,* или *структура* – это набор хромосом данной особи. Следовательно, особями популяции могут быть генотипы либо единичные хромосомы (в довольно распространенном случае, когда генотип состоит из одной хромосомы).

*Фенотип* – это набор значений, соответствующих данному генотипу, т. е. *декодированная структура,* или множество *параметров задачи (решение, точка пространства поиска).*

*Аллель* – это значение конкретного гена, также определяемое как *значение свойства,* или *вариант свойства.*

*Локус,* или *позиция,* указывает место размещения данного гена в хромосоме (цепочке). Множество позиций генов – это *локи.*

Очень важным понятием в генетических алгоритмах считается *функция приспособленности (fitness function),* иначе называемая *функцией оценки.* Она представляет меру приспособленности данной особи в популяции. Эта функция играет важнейшую роль, поскольку позволяет оценить степень приспособленности конкретных особей в популяции и выбрать из них наиболее приспособленные (имеющие наибольшие значения функции приспособленности) в соответствии с эволюционным принципом выживания сильнейших (лучше всего приспособившихся). Функция приспособленности также получила свое название непосредственно из генетики. Она оказывает сильное влияние на функционирование генетических алгоритмов и должна иметь точное и корректное определение. В задачах оптимизации функция приспособленности, как правило, оптимизируется (точнее говоря, максимизируется) и называется *целевой функцией.* В задачах минимизации целевая функция преобразуется и проблема сводится к максимизации. В теории управления функция приспособленности может принимать вид *функции погрешности,* а в теории игр – *стоимостной функции.* На

каждой итерации генетического алгоритма приспособленность каждой особи данной популяции оценивается при помощи функции приспособленности, и на этой основе создается следующая популяция особей, составляющих множество потенциальных решений проблемы, например, задачи оптимизации.

Очередная популяция в генетическом алгоритме называется *поколением,* а к вновь создаваемой популяции особей применяется термин «новое поколение» или *поколение потомков*.

***Пример***

Рассмотрим функцию

$$f(x) = 2x^2 + 1 \qquad (6.1)$$

и допустим, что x принимает целые значения из интервала от 0 до 15. Задача оптимизации этой функции заключается в перемещении по пространству, состоящему из 16 точек со значениями 0, 1, ..., 15 для обнаружения той точки, в которой функция принимает максимальное (или минимальное) значение.

Разумеется, решение этой задачи с позиций обычной алгебры совершенно очевидно. Максимальное значение функция достигает при $x = 15$. Тем интереснее проследить, как с этой задачей справляется генетический алгоритм, которому не известны математические правила.

В этом случае в качестве *параметра задачи* выступает переменная $x$. Множество {0, 1, ..., 15} составляет *пространство поиска* и одновременно – множество потенциальных решений задачи. Каждое из 16 чисел, принадлежащих к этому множеству, называется *точкой пространства поиска, решением, значением параметра, фенотипом.* Следует отметить, что решение, оптимизирующее функцию, называется *наилучшим,* или *оптимальным* решением. Значения параметра $x$ от 0 до 15 можно закодировать следующим образом:

0000 0001 0010 0011 0100 0101 0110 0111

1000 1001 1010 1011 1100 1101 1110 1111

Это широко известный способ двоичного кодирования, связанный с записью десятичных цифр в двоичной системе. Представленные *кодовые последовательности* также называются *цепями,* или хромосомами. В рассматриваемом примере они выступают и в роли *генотипов.* Каждая из хромосом состоит из 4 генов (иначе можно сказать, что двоичные последовательности состоят из 4 битов). Значение гена в конкретной позиции называется аллелью, принимающей в данном случае значения 0 или 1. *Популяция* состоит из особей, выбираемых среди этих 16 хромосом. Примером популяции с численностью, равной 6, может быть, например, множество хромосом {0010, 0101, 0111, 1001, 1100, 1110}, представляющих собой закодированную форму следующих фенотипов: {2, 5, 7, 9, 12, 14}. *Функция приспособленности в* этом примере задается выражением (6.1). Приспособленность отдельных хромосом в популяции определяется значением этой функции для значений x, соответствующих данным хромосомам, т. е. для *фенотипов,* соответствующих определенным *генотипам.*

### *6.4. Классический генетический алгоритм*

Основной (классический) генетический алгоритм (также называемый элементарным, или простым, генетическим алгоритмом) состоит из следующих шагов:

1) инициализации, или выбора исходной популяции хромосом;
2) оценки приспособленности хромосом в популяции;
3) проверки условия остановки алгоритма;
4) селекции хромосом;
5) применения генетических операторов;
6) формирования новой популяции;
7) выбора «наилучшей» хромосомы.

Блок-схема основного генетического алгоритма изображена на рис. 6.2.

Рассмотрим конкретные этапы данного алгоритма.

*Инициализация*, т. е. формирование исходной популяции заключается в случайном выборе заданного количества хромосом (особей), представляемых двоичными последовательностями фиксированной длины.

*Оценивание приспособленности* хромосом в популяции состоит в расчете функции приспособленности для каждой хромосомы этой популяции. Чем больше значение этой функции, тем выше «качество» хромосомы. Форма функции приспособленности зависит от характера решаемой задачи. Предполагается, что функция приспособленности всегда принимает неотрицательные значения и, кроме того, для решения оптимизационной задачи требуется максимизировать эту функцию. Если исходная форма функции приспособленности не удовлетворяет данным условиям, то выполняется соответствующее преобразование (например, задачу минимизации функции можно легко свести к задаче максимизации).

*Проверка условия остановки алгоритма*. Определение условия остановки генетического алгоритма зависит от его конкретного применения. В оптимизационных задачах, если известно максимальное (или минимальное) значение функции приспособленности, остановка алгоритма может произойти после достижения ожидаемого оптимального значения, возможно, с заданной точностью. Остановка алгоритма также может произойти в случае, когда его выполнение не приводит к улучшению уже достигнутого значения. Алгоритм может быть остановлен по истечении определенного времени выполнения либо после выполнения заданного количества итераций. Если условие остановки выполнено, то производится переход к завершающему этапу выбора «наилучшей» хромосомы. В противном случае на следующем шаге выполняется селекция.

*Селекция хромосом* заключается в выборе (по рассчитанным на втором этапе значениям функции приспособленности) тех хромосом, которые будут участвовать в создании потомков для следующей популяции, т. е. для очередного поколения. Такой выбор производится согласно принципу естественного отбора, по которому наибольшие шансы на участие в создании новых особей имеют

хромосомы с наибольшими значениями функции приспособленности. Существуют различные методы селекции.

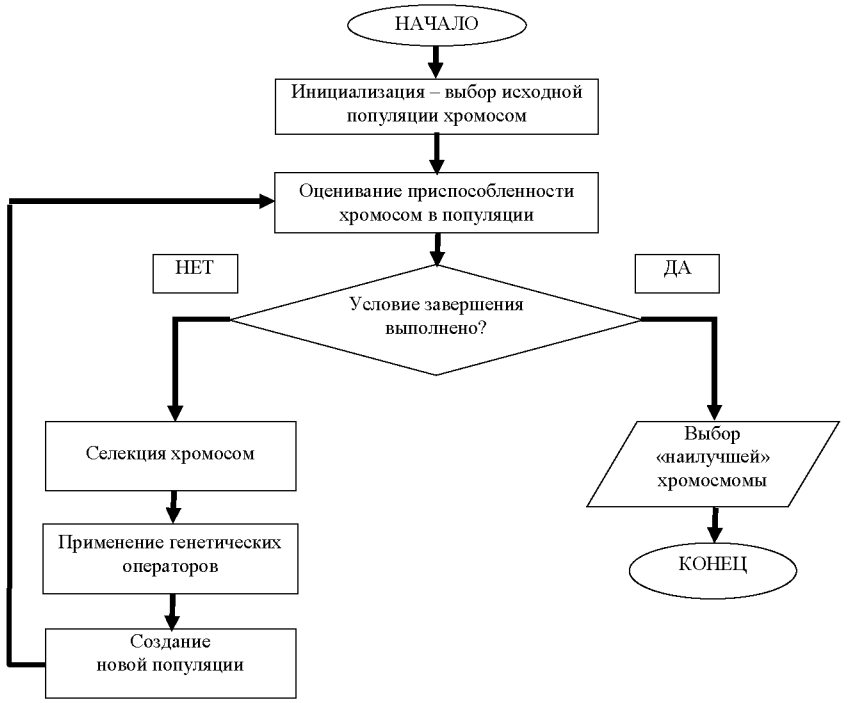



Рис. 6.2

Наиболее популярным считается так называемый метод *рулетки (roulette wheel selection),* который свое название получил по аналогии с известной азартной игрой. Каждой хромосоме может быть сопоставлен сектор колеса рулетки, значения которого устанавливается пропорциональной значению функции приспособленности данной хромосомы. Поэтому чем больше значение функции приспособленности, тем больше сектор на колесе рулетки. Все колесо рулетки соответствует сумме значений функции приспособленности всех хромосом рассматриваемой популяции. Каждой хромосоме, обозначаемой $ch_i$ для $I = 1,2\ N$ (где $N$ обозначает численность

популяции), соответствует сектор колеса $v(ch_i)$, выраженный в процентах согласно формуле

$$v(ch_i) = p_s(ch_i)\ 100\ \%, \qquad (6.2)$$

где

$$p_s(ch_i) = \frac{F(ch_i)}{\sum_{i=1}^{N} F(ch_i)}, \qquad (6.3)$$

причем $F(ch_i)$ – значение функции приспособленности хромосомы $ch_i$, а $p_s(ch_i)$ – *вероятность селекции* хромосомы $ch_i$. Селекция хромосомы может быть представлена как результат поворота колеса рулетки, поскольку «выигравшая» (т. е. выбранная) хромосома относится к выпавшему сектору этого колеса. Очевидно, что чем больше сектор, тем больше вероятность «победы» соответствующей хромосомы. Поэтому вероятность выбора данной хромосомы оказывается пропорциональной значению ее функции приспособленности. Если всю окружность колеса рулетки представить в виде цифрового интервала [0, 100], то выбор хромосомы можно отождествить с выбором числа из интервала [$a$, $b$], где $a$ и $b$ обозначают соответственно начало и окончание фрагмента окружности, соответствующего этому сектору колеса; очевидно, что $0 < a < b < 100$. В таком случае выбор с помощью колеса рулетки сводится к выбору числа из интервала [0, 100], которое соответствует конкретной точке на окружности колеса.

В результате процесса селекции создается *родительская популяция,* также называемая *родительским пулом (mating pool)* с численностью $N$, равной численности текущей популяции.

*Применение генетических операторов* к хромосомам, отобранным с помощью селекции, приводит к формированию новой популяции потомков от созданной на предыдущем шаге родительской популяции.

В классическом генетическом алгоритме применяется два основных генетических оператора: *оператор скрещивания (crossover)* и *оператор мутации (mutation)*. Однако следует отметить, что оператор мутации играет явно

второстепенную роль по отношению к оператору скрещивания. Это означает, что скрещивание в классическом генетическом алгоритме производится практически всегда, тогда как мутация – достаточно редко. Вероятность скрещивания, как правило, достаточно велика (обычно $0{,}5 < p_c < 1$), тогда как вероятность мутации устанавливается весьма малой (чаще всего $0 < p_m < 0{,}1$). Это следует из аналогии с миром живых организмов, где мутации происходят чрезвычайно редко.

В генетическом алгоритме мутация хромосом может выполняться на популяции родителей перед скрещиванием либо на популяции потомков, образованных в результате скрещивания.

*Оператор скрещивания*. На первом этапе скрещивания выбираются пары хромосом из родительской популяции (родительского пула). Это временная популяция, состоящая из хромосом, отобранных в результате селекции и предназначенных для дальнейших преобразований операторами скрещивания и мутации для формирования новой популяции потомков. На данном этапе хромосомы из родительской популяции объединяются в пары. Это происходят случайным способом в соответствии с вероятностью скрещивания $p_c$. Далее для каждой пары отобранных таким образом родителей разыгрывается позиция гена *(локус)* в хромосоме, определяющая так называемую *точку скрещивания.* Если хромосома каждого из родителей состоит из $L$ генов, то точка скрещивания $I_k$ представляет собой натуральное число, меньшее $L$. Поэтому фиксация точки скрещивания сводится к случайному выбору числа из интервала $[1, L\text{-}1]$. В результате скрещивания пары родительских хромосом получается следующая пара потомков:

1) потомок, хромосома которого на позициях от 1 до $I_k$ состоит из генов первого родителя, а на позициях от $I_k + 1$ до $L$ – из генов второго родителя;

2) потомок, хромосома которого на позициях от 1 до $I_k$ состоит из генов второго родителя, а на позициях от $I_k$+1 до $L$ – из генов первого родителя.

*Оператор мутации* с вероятностью $p_m$ изменяет значение гена в хромосоме на противоположное (т. е. с 0 на 1 или обратно). Например, если в

хромосоме [100110101010] мутации подвергается ген на позиции 7, то его значение, равное 1, изменяется на 0, что приводит к образованию хромосомы [100110001010]. Как уже упоминалось выше, вероятность мутации обычно очень мала, и именно от нее зависит, будет ли данный ген подвержен мутации или нет. Вероятность $p_m$ мутации может эмулироваться, например, случайным выбором числа из интервала [0, 1] для каждого гена и отбором для выполнения этой операции тех генов, для которых разыгранное число оказывается меньшим или равным значению $p_m$.

*Формирование новой популяции*. Хромосомы, полученные в результате применения генетических операторов к хромосомам временной родительской популяции, включаются в состав новой популяции. Она становится так называемой текущей популяцией для данной итерации генетического алгоритма. На каждой очередной итерации рассчитываются значения функции приспособленности для всех хромосом этой популяции, после чего проверяется условие остановки алгоритма и либо фиксируется результат в виде хромосомы с наибольшим значением функции приспособленности, либо осуществляется переход к следующему шагу генетического алгоритма, т. е. к селекции. В классическом генетическом алгоритме вся предшествующая популяция хромосом замещается новой популяцией потомков, имеющей ту же численность.

*Выбор «наилучшей» хромосомы*. Если условие остановки алгоритма выполнено, то следует вывести результат работы, т. е. представить искомое решение задачи. Лучшим решением считается хромосома с наибольшим значением функции приспособленности.

Следует отметить, что генетические алгоритмы унаследовали свойства естественного эволюционного процесса, состоящие в генетических изменениях популяций организмов с течением времени.

Главный фактор эволюции – это естественный отбор (природная селекция), который приводит к тому, что среди генетически различающихся особей одной и той же популяции выживают и оставляют потомство только

наиболее приспособленные к окружающей среде. В генетических алгоритмах также выделяется этап селекции, на котором из текущей популяции выбираются и включаются в родительскую популяцию особи, имеющие наибольшие значения функции приспособленности. На следующем этапе, который иногда называется *эволюцией,* применяются генетические операторы скрещивания и мутации, выполняющие рекомбинацию генов в хромосомах.

Операция скрещивания заключается в обмене фрагментами цепочек между двумя родительскими хромосомами. Пары родителей для скрещивания выбираются из родительского пула случайным образом так, чтобы вероятность выбора конкретной хромосомы для скрещивания была равна вероятности $p_c$. Например, если в качестве родителей случайным образом выбираются две хромосомы из родительской популяции численностью $N$, то $p_c = 2/N$. Аналогично, если из родительской популяции численностью $N$ выбирается $2z$ хромосом
($z < N/2$), которые образуют $z$ пар родителей, то $p_c = 2z/N$. Обратим внимание, что если все хромосомы текущей популяции объединены в пары до скрещивания, то $p_c = 1$. После операции скрещивания родители в родительской популяции замещаются их потомками.

Операция мутации изменяет значения генов в хромосомах с заданной вероятностью $p_m$ способом, представленным при описании соответствующего оператора. Это приводит к инвертированию значений отобранных генов с 0 на 1 и обратно. Значение $p_m$, как правило, очень мало, поэтому мутации подвергается лишь небольшое количество генов. Скрещивание – это ключевой оператор генетических алгоритмов, определяющий их возможности и эффективность. Мутация играет более ограниченную роль. Она вводит в популяцию некоторое разнообразие и предупреждает потери, которые могли бы произойти вследствие исключения какого-нибудь значимого гена в результате скрещивания.

Основной (классический) генетический алгоритм известен в качестве инструмента, в котором выделяется три вида операций: *репродукции, скрещивания* и *мутации.* Термины *селекция* и *репродукция* в данном контексте

используются в качестве синонимов. При этом репродукция в данном случае связывается с созданием копий хромосом родительского пула, тогда как более широкое значение данного понятия процесс формирования новых особей, происходящих от конкретных родителей. Если мы принимаем такое толкование, то операторы скрещивания и мутации могут считаться операторами репродукции, а селекция – отбором особей (хромосом) для репродукции.

### *6.5. Иллюстрация выполнения классического генетического алгоритма*

Рассмотрим выполнение описанного в предыдущем пункте классического генетического алгоритма на как можно более простом примере. Проследим последовательность выполнения его этапов, соответствующих блок-схеме на рис. 6.1.

Рассмотрим сильно упрощенный и довольно искусственный пример, состоящий в нахождении хромосомы с максимальным количеством единиц. Допустим, что хромосомы состоят из 12 генов, а популяция насчитывает 8 хромосом. Наилучшей в таком случае будет хромосома, состоящая из 12 единиц. Посмотрим, как протекает процесс решения этой весьма тривиальной задачи с помощью генетического алгоритма.

**Инициализация, или выбор исходной популяции хромосом.** Необходимо случайным образом сгенерировать 8 двоичных последовательностей длиной 12 битов. Это можно достигнуть, например, подбрасыванием монеты 96 раз (при выпадении «орла» приписывается значение 1, а в случае «решки» 0). Таким образом, можно сформировать исходную популяцию

$ch_1$ = [111001100101]; $ch_5$ = [010001100100];

$ch_2$ = [001100111010]; $ch_6$ = [010011000101];

$ch_3$ = [011101110011]; $ch_7$ = [101011011011];

$ch_4$ = [001000101000]; $ch_8$ = [000010111100] .

**Оценка приспособленности хромосом в популяции.** В рассматриваемом упрощенном примере решается задача нахождения такой хромосомы, которая содержит наибольшее количество единиц, поэтому функция приспособленности определяет количество единиц в хромосоме. Обозначим функцию приспособленности символом $F$, тогда ее значения для каждой хромосомы из исходной популяции будут такие:

$F(ch_1) = 7$; $F(ch_5) = 4$;

$F(ch_2) = 6$; $F(ch_6) = 5$;

$F(ch_3) = 8$; $F(ch_7) = 8$;

$F(ch_4) = 3$; $F(ch_8) = 5$ .

Хромосомы $ch_3$ и $ch_7$ имеют наибольшие значения функции принадлежности. В этой популяции они считаются наилучшими кандидатами на решение задачи. Если в соответствии с блок-схемой генетического алгоритма (см.рис.6.1) условие остановки алгоритма не выполняется, то на следующем шаге производится селекция хромосом из текущей популяции.

**Селекция хромосом.** Селекция производится методом рулетки. На основании формул (6.2) и (6.3) для каждой из 8 хромосом популяции (в нашем случае исходной популяции, для которой $N = 8$) получаем секторы колеса рулетки, выраженные в процентах (рис. 6.2):

$v(ch_1) = 15{,}22$; $v(ch_5) = 8{,}70$;

$v(ch_2) = 13{,}04$; $v(ch_6) = 10{,}87$;

$v(ch_3) = 17{,}39$; $v(ch_7) = 17{,}39$;

$v(ch_4) = 6{,}52$; $v(ch_8) = 10{,}87$.

Розыгрыш с помощью колеса рулетки сводится к случайному выбору числа из интервала [0, 100], указывающего на соответствующий сектор колеса, т. е. на конкретную хромосому. Допустим, что были разыграны следующие 8 чисел:

79 44 9 74 44 86 48 23.

Это означает выбор хромосом

$ch_7\ ch_3\ ch_1\ ch_7\ ch_3\ ch_7\ ch_4\ ch_2$.

Как видно, хромосома $ch_7$ была выбрана трижды, а хромосома $ch_3$ – дважды. Заметим, что именно эти хромосомы имеют наибольшее значение функции приспособленности. Однако выбрана и хромосома $ch_4$ с наименьшим значением функции приспособленности. Все выбранные таким образом хромосомы включаются в так называемый родительский пул.

**Применение генетических операторов.** Допустим, что ни одна из отобранных в процессе селекции хромосом не подвергается мутации, и все они составляют популяцию хромосом, предназначенных для скрещивания. Это означает, что вероятность скрещивания $p_c = 1$, а вероятность мутации $p_m = 0$.

Допустим, что из этих хромосом случайным образом сформированы пары родителей

$ch_2$ и $ch_7$

$ch_1$ и $ch_7$

$ch_3$ и $ch_4$

$ch_3$ и $ch_7$

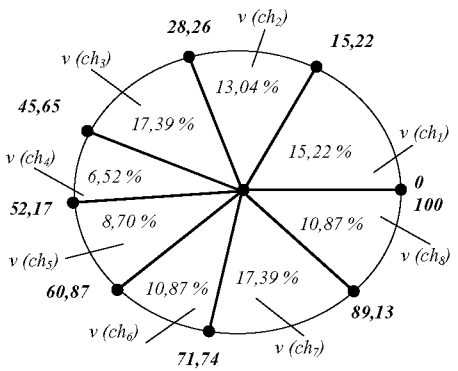



Рис. 6.3

Для первой пары случайным образом выбрана точка скрещивания – $I_k = 4$, для второй – $I_k = 3$, для третьей – $I_k = 11$, для четвертой – $I_k = 5$. При этом процесс скрещивания протекает так, как показано на рис. 6.3. В результате выполнения оператора скрещивания получаются 4 пары потомков.

Если бы при случайном подборе пар хромосом для скрещивания были объединены, например, $ch_3$ с $ch_3$ и $ch_4$ с $ch_7$ вместо $ch_3$ с $ch_4$ и $ch_3$ с $ch_7$, а другие

пары остались без изменения, то скрещивание $ch_3$ с $ch_3$ дало бы две такие же хромосомы независимо от разыгранной точки скрещивания. Это означало бы получение двух потомков, идентичных своим родителям. Заметим, что такая ситуация наиболее вероятна для хромосом с наибольшим значением функции приспособленности, т. е. именно такие хромосомы получают наибольшие шансы на переход в новую популяцию.

**Формирование новой популяции.** После выполнения операции скрещивания мы получаем следующую популяцию потомков:

$Ch_1$ = [001111011011];
$Ch_2$ = [101000111010];
$Ch_3$ = [111011011011];
$Ch_4$ = [101001100101];
$Ch_5$ = [011101110010];
$Ch_6$ = [001000101001];
$Ch_7$ = [011101011011];
$Ch8$ = [101011110011].

Для отличия от хромосом предыдущей популяции обозначения вновь сформированных хромосом начинаются с заглавной буквы С.

Согласно блок-схеме генетического алгоритма (рис. 6.1) производится возврат ко второму этапу, т. е. к оценке приспособленности хромосом из вновь сформированной популяции, которая становится текущей. Значения функций приспособленности хромосом этой популяции составляют

$F(Ch_1) = 8$;
$F(Ch_2) = 6$;
$F(Ch_3) = 9$;
$F(Ch_4) = 6$;
$F(Ch_5) = 7$;
$F(Ch_6) = 4$;
$F(Ch_7) = 8$;
$F(Ch_8) = 8$.

Заметно, что популяция потомков характеризуется гораздо более высоким средним значением функции приспособленности, чем популяция родителей. Обратим внимание, что в результате скрещивания получена хромосома $Ch_3$ с наибольшим значением функции приспособленности, которым не обладала ни одна хромосома из родительской популяции. Однако могло произойти и обратное, поскольку после скрещивания на первой итерации хромосома, которая в родительской популяции характеризовалась наибольшим значением функции приспособленности, могла просто «потеряться». Помимо этого «средняя»

приспособленность новой популяции все равно оказалась бы выше предыдущей, а хромосомы с большими значениями функции приспособленности имели бы шансы появиться в следующих поколениях.

Рассмотрим очень простой рассмотренный выше пример – задачу нахождения максимума функции, которая задана выражением (6.1) для целочисленной переменной $x$, принимающей значения от 0 до 31.

Для применения генетического алгоритма необходимо прежде всего закодировать значения переменной $x$ в виде двоичных последовательностей. Очевидно, что целые числа из интервала [0, 31] можно представить последовательностями нулей и единиц, используя их представление в двоичной системе счисления. Число 0 при этом записывается как 00000, а число 31 – как 11111. В данном случае хромосомы приобретают вид двоичных последовательностей, состоящих из 5 битов, т. е. цепочками длиной 5 (аналогично примеру 6.1).

В качестве функции приспособленности будет выступать целевая функция $f(x)$, заданная выражением (6.1). Тогда приспособленность хромосомы $ch_i$, $i = 1,2, ..., N$ будет определяться значением функции $f(x)$ для $x$, равного фенотипу, соответствующему генотипу $ch_i$. Обозначим эти фенотипы $ch_i$*. В таком случае значение функции приспособленности хромосомы $ch_i$- (т. е. $F(ch_i)$) будет равно $f(ch_i*)$.

Выберем случайным образом исходную популяцию, состоящую из 6 кодовых последовательностей (например, можно 30 раз подбросить монету); при этом $N = 6$. Допустим, что выбраны хромосомы:

$ch_1 = [10011]$; $ch_4 = [10101]$;

$ch_2 = [00011]$; $ch_5 = [01000]$;

$ch_3 = [00111]$; $ch_6 = [11101]$.

Соответствующие им фенотипы – это представленные ниже числа из интервала от 0 до 31:

$ch_1* = 19$; $ch_4* = 21$;

$ch_2* = 3$; $ch_5* = 8$;

$ch_3$* = 7; $ch_6$* = 29 .

По формуле (6.1) рассчитываем значения функции приспособленности для каждой хромосомы в популяции и получаем:

$F(ch_1)$ = 723; $F(ch_4)$ = 883;

$F(ch_2)$ = 19; $F(ch_5)$ = 129;

$F(ch_3)$ = 99; $F(ch_6)$ = 1683.

**Селекция хромосом.** Методом рулетки выбираем 6 хромосом для репродукции. Колесо рулетки представлено на рис. 6.4.

Допустим, что выбраны числа: 97, 26, 54, 13, 31, 88.

Это означает выбор хромосом

$ch_6$, $ch_4$, $ch_6$, $ch_1$, $ch_4$, $ch_6$.

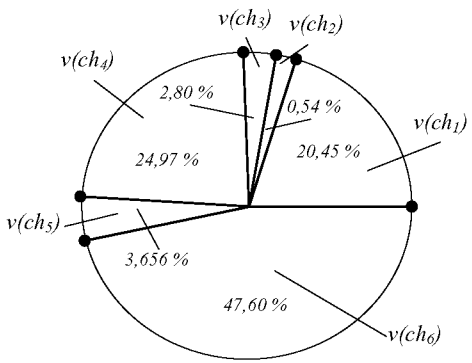



Рис. 6.4

Пусть скрещивание выполняется с вероятностью $p_c$ = 1. Допустим, что для скрещивания сформированы пары: $ch_1$ и $ch_4$ , $ch_4$ и $ch_6$ , $ch_6$ и $ch_6$ .

Кроме того, допустим, что случайным образом выбрана точка скрещивания, равная 3 для хромосом $ch_1$ и $ch_4$, а также точка скрещивания, равная 2 для хромосом $ch_4$ и $ch_6$. При условии, что вероятность мутации $p_m$ = 0, в новую популяцию включаются хромосомы

$ch_1$ = [10001]; $ch_4$ = [11101];

$ch_2$ = [10111]; $ch_5$ = [11101];

$ch_3$ = [10101]; $ch_6$ = [11101] .

Для расчета значений функции приспособленности этих хромосом необходимо декодировать представляющие их двоичные последовательности и получить соответствующие им фенотипы. Обозначим их $ch_i$*. В результате декодирования получаем числа (из интервала от 0 до 31):

$ch_1$* = 17; $ch_4$* = 29;

$ch_2$* = 23; $ch_5$* = 29;

$ch_3$* = 21; $ch_6$* = 29

Таким образом, значения функции приспособленности хромосом новой популяции, рассчитанные по формуле (6.1), составят:

$F(ch_1)$ = 579; $F(ch_4)$ = 1683;

$F(ch_2)$ = 1059; $F(ch_5)$ = 1683;

$F(ch_3)$ = 883; $F(ch_6)$ = 1683 .

Легко заметить, что в этом случае среднее значение приспособленности возросло с 589 до 1262.

Если на следующей итерации будут сформированы для скрещивания пары хромосом, например, $ch_4$ и $ch_2$, $ch_5$ и $ch_2$ или $ch_6$ и $ch_2$ с точкой скрещивания 2 или 3, то среди прочих будет получена хромосома [11111] с фенотипом, равным числу 31, при котором оптимизируемая функция достигает своего максимума. Значение функции приспособленности для этой хромосомы оказывается наибольшим и составляет 1923. Если такое сочетание пар в данной итерации не произойдет, то можно будет ожидать образования хромосомы с наибольшим значением функции приспособленности на следующих итерациях. Хромосома [11111] могла быть получена и на текущей итерации в случае формирования для скрещивания пары $ch_1$ и $ch_6$ с точкой скрещивания 3.

Отметим, что при длине хромосом, равной 5 битам, пространство поиска очень мало и насчитывает всего $2^5$ – 32 точки. Представленный пример имеет исключительно демонстрационный характер. Применение генетического алгоритма для такого простого примера нецелесообразно, поскольку его

оптимальное решение может быть получено мгновенно. Однако этот пример пригоден для изучения функционирования генетического алгоритма.

Также следует упомянуть, что в малых популяциях часто встречаются ситуации, когда на начальном этапе несколько особей имеют значительно большие значения функции принадлежности, чем остальные особи данной популяции. Применение метода селекции на основе «колеса рулетки» позволяет в этом случае очень быстро выбрать «наилучшие» особи, иногда - на протяжении «жизни» одного поколения. Однако такое развитие событий считается нежелательным, поскольку оно становится главной причиной преждевременной сходимости алгоритма, называемой сходимостью к неоптимальному решению. По этой причине используются и другие методы селекции, отличающиеся от колеса рулетки, либо применяется масштабирование функции приспособленности.

## 7. Методы прогнозирования

До недавнего времени (середины 80-х годов прошлого века) существовало несколько общепризнанных методов прогнозирования временных рядов:

- эконометрические;
- регрессионные;
- методы Бокса-Дженкинса (ARIMA, ARMA).

Однако начиная с конца 80-х годов, в научной литературе был опубликован ряд статей по нейросетевой тематике, в которых был приведен эффективный алгоритм обучения нейронных сетей и доказана возможность их использования для самого широкого круга задач. Эти статьи возродили интерес к нейросетям, которые очень скоро стали широко использоваться при исследованиях в самых разных областях науки от экспериментальной физики и химии до экономики.

Отчасти из-за относительной сложности нейронных сетей и генетических алгоритмов данные технологии не сразу вышли за рамки чисто научного применения. Тем не менее с течением времени уровень доверия к новым технологиям повышался и со стороны бизнеса. С начала 90-х годов начали регулярно появляться сообщения об установках нейросетевых систем в разных компаниях, банках, корпоративных институтах. Причем сфера использования новых технологий была очень многогранной – оценка рисков, контроль технологических процессов, управление роботами и многое другое. Одним из самых успешных приложений нейронных сетей было прогнозирование временных рядов. Причем самым массовым было: прогнозирование на финансовых рынках и прогнозирование продаж.

В настоящее время можно с уверенностью сказать, что использование нейронных сетей при прогнозировании дает ощутимое преимущество по сравнению с более простыми статистическими методами.

## *7.1. Традиционные методы прогнозирования*

### 7.1.1. "Наивные" модели прогнозирования

При создании "наивных" моделей предполагается, что некоторый последний период прогнозируемого временного ряда лучше всего описывает будущее этого прогнозируемого ряда, поэтому в этих моделях прогноз, как правило, является очень простой функцией от значений прогнозируемой переменной в недалеком прошлом.

Самой простой моделью является $Y(t + 1) = Y(t)$, что соответствует предположению, что "завтра будет как сегодня". Вне всякого сомнения, от данной примитивной модели не стоит ждать большой точности. Она не только не учитывает механизмы, определяющие прогнозируемые данные (этот серьезный недостаток вообще свойствен многим статистическим методам прогнозирования), но и не защищена от случайных флуктуаций, модель не

учитывает сезонные колебания и тренды. Впрочем, можно строить "наивные" модели несколько по-другому:

$$Y(t + 1) = Y(t) + [Y(t) - Y(t-1)],$$

$$Y(t + 1) = Y(t) * [Y(t)/Y(t-1)].$$

Такими способами мы пытаемся приспособить модель к возможным трендам

$$Y(t + 1) = Y(t-s),$$

это попытка учесть сезонные колебания.

### 7.1.2. Средние и скользящие средние

Самой простой моделью, основанной на *простом усреднении,* является следующая:

$$Y(t + 1) = (1/(t)) * [Y(t) + Y(t - 1) + ... + Y(1)] .$$

В отличии от самой простой "наивной" модели, которой соответствовал принцип "завтра будет, как сегодня", этой модели соответствует принцип "завтра будет как было в среднем за последнее время". Такая модель, конечно, более устойчива к флуктуациям, поскольку в ней сглаживаются случайные выбросы относительно среднего. Несмотря на это, данный метод идеологически настолько же примитивен как и "наивные" модели и ему свойственны почти те же самые недостатки.

В приведенной выше формуле подразумевалось, что ряд усредняется по достаточно длительному интервалу времени. Однако, как правило, значения временного ряда из недалекого прошлого лучше описывают прогноз, чем более старые значения этого же ряда. Тогда можно использовать для прогнозирования *скользящее среднее*:

$$Y(t + 1)=(1/(T + 1))*[Y(t) + Y(t-1) + ... + Y(t-T)] .$$

Смысл его заключается в том, что модель видит только ближайшее прошлое (на $T$ отсчетов по времени в глубину) и, основываясь только на этих данных, строит прогноз.

При прогнозировании довольно часто используется метод *экспоненциальных средних*, который постоянно адаптируется к данным за счет новых значений. Формула, описывающая эту модель, выглядит следующим образом:

$$Y(t+1) = a * Y(t) + (1-a) * \hat{Y}(t),$$

где $Y(t+1)$ – прогноз на следующий период времени; $Y(t)$ – реальное значение в момент времени $t$; $\hat{Y}(t)$ – прошлый прогноз на момент времени $t$; $a$ – постоянная сглаживания ( $0 \le a \le 1$ ).

В этом методе есть внутренний параметр $a$, который определяет зависимость прогноза от более старых данных, причем влияние данных на прогноз убывает с "возрастом" данных. Зависимость влияния данных на прогноз при разных коэффициентах $a$ приведена на графике (см.рис.7.1).

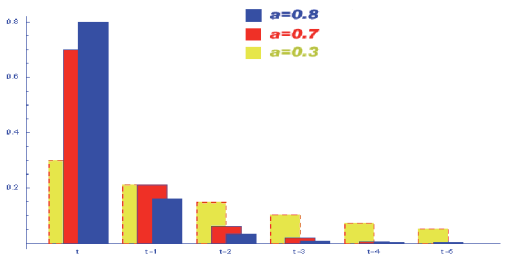



Рис. 7.1

Видно, что при $a = 1$, экспоненциальная модель стремится к самой простой "наивной" модели. При $a = 0$ прогнозируемая величина становится равной предыдущему прогнозу.

Если производится прогнозирование с использованием модели экспоненциального сглаживания, обычно на некотором тестовом наборе строятся прогнозы при $a$ = [0,01; 0,02; ...; 0,98; 0,99] и отслеживается, при каком $a$ точность прогнозирования выше. Это значение $a$ затем используется при прогнозировании в дальнейшем.

Описанные выше модели ("наивные" алгоритмы, методы, основанные на средних, скользящих средних и экспоненциального сглаживания) могут быть использованы при прогнозировании бизнес-процессов в самых простых

ситуациях и не рекомендуются в общем случае из-за своей простоты и неадекватности моделей. Вместе с тем описанные алгоритмы вполне успешно можно использовать как сопутствующие и вспомогательные для предобработки данных в задачах прогнозирования. Например, для прогнозирования продаж в большинстве случаев необходимо проводить декомпозицию временных рядов (т. е. выделять отдельно тренд, сезонную и нерегулярную составляющие). Одним из методов выделения трендовых составляющих является использование экспоненциального сглаживания.

### 7.1.3. Методы Хольта и Брауна

В XXв. Хольт предложил усовершенствованный метод экспоненциального сглаживания, впоследствии названный его именем. В предложенном алгоритме значения уровня и тренда сглаживаются с помощью экспоненциального сглаживания, причем параметры сглаживания у них различны

$$\Omega_t = \alpha Y_t + (1-\alpha)(\Omega_{t-1} - T_{t-1}) \tag{7.2}$$

$$T_t = \beta(\Omega_t - \Omega_{t-1}) + (1-\beta)T_{t-1} \tag{7.3}$$

$$\hat{Y}_{t+p} = \Omega_t + pT \tag{7.4}$$

Здесь уравнение 7.2 описывает сглаженный ряд общего уровня; 7.3 уравнение служит для оценки тренда; 7.4 уравнение определяет прогноз на $p$ отсчетов по времени вперед.

Постоянные сглаживания в методе Хольта играют ту же роль, что и постоянная в простом экспоненциальном сглаживании. Находится они, например, путем перебора по этим параметрам с каким-то шагом. Можно использовать и менее сложные в отношении количества вычислений алгоритмы. Главное, что всегда можно подобрать такую пару параметров, которая дает большую точность модели на тестовом наборе и затем использовать эту пару параметров при реальном прогнозировании.

Частным случаем метода Хольта является метод Брауна, при котором $a$ = ß.

### 7.1.4. Метод Винтерса

Хотя описанный выше метод Хольта (метод двухпараметрического экспоненциального сглаживания) и не является совсем простым (относительно "наивных" моделей и моделей, основанных на усреднении), он не позволяет учитывать сезонные колебания при прогнозировании. Говоря более аккуратно, этот метод не может их "видеть" в предыстории. Существует расширение метода Хольта до трехпараметрического экспоненциального сглаживания. Такой алгоритм называется методом Винтерса. При этом делается попытка учесть сезонные составляющие в данных. Система уравнений, описывающих метод Винтерса, выглядит следующим образом:

$$\Omega_t = \alpha \frac{Y_t}{S_{t-s}} + (1-\alpha)(\Omega_{t-1} - T_{t-1}),$$
$$T_t = \beta(\Omega_t - \Omega_{t-1}) + (1-\beta)T_{t-1},$$
$$S_t = \gamma \frac{Y_t}{\Omega_t} + (1-\gamma)S_{t-s},$$
$$\hat{Y}_{t+p} = (\Omega_t + pT_t)S_{t-s+p}$$

Дробь в первом уравнении служит для исключения сезонности из *Y(t)*. После исключения сезонности алгоритм работает с чистыми данными, в которых нет сезонных колебаний. Появляются они уже в самом финальном прогнозе, когда "чистый" прогноз, посчитанный по методу Хольта, умножается на сезонный коэффициент.

### 7.1.5. Регрессионные методы прогнозирования

Наряду с описанными выше методами, основанными на экспоненциальном сглаживании, уже достаточно долгое время для прогнозирования используются регрессионные алгоритмы. Кратко суть алгоритмов такого класса можно описать так.

Существует прогнозируемая переменная *Y* (зависимая переменная) и отобранный заранее комплект переменных, от которых она зависит – *X1*, *X2*, ..., *XN* (независимые переменные). Природа независимых переменных может быть различной. Например, если предположить, что *Y* – уровень спроса на

некоторый продукт в следующем месяце, то независимыми переменными могут быть уровень спроса на этот же продукт в прошлый и позапрошлый месяцы, затраты на рекламу, уровень платежеспособности населения, экономическая обстановка, деятельность конкурентов и многое другое. Главное – уметь формализовать все внешние факторы, от которых может зависеть уровень спроса в числовую форму.

Модель множественной регрессии в общем случае описывается выражением

$$Y = F(X_1, X_2, \ldots, X_N) + \varepsilon.$$

В более простом варианте линейной регрессионной модели зависимость зависимой переменной от независимых имеет вид:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_N X_N + \ldots + \varepsilon$$

Здесь $\beta_0$, $\beta_1$, $\beta_2$,…… $\beta_N$ – подбираемые коэффициенты регрессии, $\varepsilon$ – компонента ошибки. Предполагается, что все ошибки независимы и нормально распределены.

Для построения регрессионных моделей необходимо иметь базу данных наблюдений (таблица)

| | Переменные | | | | |
|---|---|---|---|---|---|
| | **независимые** | | | | **зависимая** |
| | *X1* | *X2* | ... | *XN* | *Y* |
| 1 | x_11 | x_12 | ... | x_1N | Y_1 |
| 2 | x_21 | x_22 | ... | x_2N | Y_2 |
| ... | ... | ... | ... | ... | ... |
| m | x_M1 | x_M2 | ... | x_MN | Y_m |

Рис.7.2

С помощью таблицы значений по прошлым наблюдениям можно подобрать (например, методом наименьших квадратов) коэффициенты регрессии, настроив тем самым модель.

При работе с регрессией надо соблюдать определенную осторожность и обязательно проверять найденные модели на адекватность. Существуют разные способы такой проверки. Обязательным является статистический анализ остатков, тест Дарбина-Уотсона. Полезно, как и в случае с нейронными сетями, иметь независимый набор примеров, на которых можно проверить качество работы модели.

### 7.1.6. Методы Бокса-Дженкинса (ARIMA)

В сер. 90-х г. прошлого века был разработан принципиально новый и достаточно мощный класс алгоритмов для прогнозирования временных рядов. Большая часть работы по исследованию методологии и проверке моделей была проведена двумя статистиками: Г. Е. П. Боксом (G. E. P. Box) и Г. М. Дженкинсом (G. M. Jenkins). С тех пор построение подобных моделей и получение на их основе прогнозов иногда называется методами Бокса-Дженкинса. Отметим, что в это семейство входит несколько алгоритмов, самым известным и используемым из них является алгоритм ARIMA. Он встроен практически в любой специализированный пакет для прогнозирования.

В классическом варианте ARIMA не используются независимые переменные. Модели опираются только на информацию, содержащуюся в предыстории прогнозируемых рядов, что ограничивает возможности алгоритма. В отличие от рассмотренных ранее методик прогнозирования временных рядов, в методологии ARIMA не предполагается какой-либо четкой модели для прогнозирования данной временной серии. Задается лишь общий класс моделей, описывающих временной ряд и позволяющих как-то выражать текущее значение переменной через ее предыдущие значения. Далее алгоритм, подстраивая внутренние параметры, сам выбирает наиболее подходящую

модель прогнозирования. Как уже отмечалось выше, существует целая иерархия моделей Бокса-Дженкинса. Логически ее можно определить так:

$AR(p) + MA(q) \geq ARMA(p, q) \geq ARMA(p, q)(P, Q) \geq ARIMA(p, q, r)(P, Q, R) \geq ..$

где $AR(p)$ – авторегрессионая модель порядка p.

Модель имеет вид

$Y(t) = f_0 + f_1*Y(t-1) + f_2*Y(t-2) + ... + f_p*Y(t-p) + E(t)$ ,

где $Y(t)$ – зависимая переменная в момент времени t;

$f_0, f_1, f_2, ..., f_p$ – оцениваемые параметры;

$E(t)$ – ошибка от влияния переменных, которые не учитываются в данной модели.

Задача заключается в том, чтобы определить $f_0, f_1, f_2, ..., f_p$. Их можно оценить различными способами. Правильнее всего искать их через систему уравнений Юла-Уолкера, для составления которой потребуется расчет значений автокорреляционной функции. Можно поступить более простым способом – посчитать их методом наименьших квадратов.

$MA(q)$ – модель со скользящим средним порядка q.

Модель имеет вид:

$$Y(t) = m + e(t) - w\_1*e(t-1) - w\_2*e(t-2) - ... - w\_p*e(t-p)$$

где $Y(t)$ – зависимая переменная в момент времени $t$; $w\_0, w\_1, w\_2, ..., w\_p$ – оцениваемые параметры.

### *7.2. Нейросетевые модели бизнес-прогнозирования*

В настоящее время самым перспективным количественным методом прогнозирования является использование нейронных сетей. Можно назвать много преимуществ нейронных сетей над остальными алгоритмами, среди которых отметим два следующих.

При использовании нейронных сетей легко исследовать зависимость прогнозируемой величины от независимых переменных. Например, предположим, что продажи на следующей неделе каким-то образом зависят от следующих параметров:

- продажи в последнюю неделю;
- продажи в предпоследнюю неделю;
- время прокрутки рекламных роликов (TRP);
- количество рабочих дней;
- температура воздуха и т. д.

Кроме того, считаем, что продажи носят сезонный характер, имеют тренд и как-то зависят от активности конкурентов. Требуется построить систему, которая естественным образом учитывала бы все это и строила краткосрочные прогнозы.

В такой постановке задачи большая часть классических методов прогнозирования будет просто несостоятельной. Можно построить систему на основе нелинейной множественной регрессии, или вариации сезонного алгоритма ARIMA, позволяющей учитывать внешние параметры, но это будут, скорее всего, малоэффективные модели (за счет субъективного выбора модели) и крайне негибкие.

Используя даже самую простую нейросетевую архитектуру (персептрон с одним скрытым слоем) и базу данных (с продажами и всеми параметрами), легко получить работающую систему прогнозирования. Причем учет или неучет системой внешних параметров будет определяться включением или исключением соответствующего входа в нейронную сеть.

Эксперт может с самого начала воспользоваться каким-либо алгоритмом определения важности (например, используя нейронную сеть с общей регрессией и генетической подстройкой) и сразу определить значимость входных переменных, чтобы потом исключить из рассмотрения мало влияющие параметры.

Еще одно важное преимущество нейронных сетей состоит в том, что эксперт не является заложником выбора математической модели поведения временного ряда. Построение нейросетевой модели происходит адаптивно во время обучения без участия эксперта. При этом нейронной сети

предоставляются примеры из базы данных, и она сама подстраивается под эти данные.

Недостатком нейронных сетей является отсутствие объяснения принципа их функционирования. После обучения появляется "черный ящик", который каким-то образом работает, но логика принятия решений нейросетью совершенно скрыта от эксперта. Существуют алгоритмы "извлечения знаний из нейронной сети", которые формализуют обученную нейронную сеть до списка логических правил, тем самым создавая на основе сети экспертную систему. К сожалению, эти алгоритмы не встраиваются в нейросетевые пакеты, к тому же наборы правил, которые генерируются такими алгоритмами, достаточно объемные.

Тем не менее для людей, умеющих работать с нейронными сетями и знающими нюансы настройки, обучения и применения, в практических задачах непрозрачность нейронных сетей не является сколько-нибудь серьезным препятствием.

### *7.3. Использование многослойных персептронов*

Самый простой вариант применения искусственных нейронных сетей в задачах бизнес-прогнозирования – использование обычного персептрона с одним, двумя или в крайнем случае тремя скрытыми слоями. При этом на входы нейронной сети обычно подается набор параметров, на основе которого (по мнению эксперта) можно успешно прогнозировать. Выходом ее обычно является прогноз сети на будущий момент времени.

Рассмотрим пример прогнозирования продаж. На рисунке представлен график, отражающий историю продаж некого продукта по неделям. В данных заметна выраженная сезонность. Для простоты предположим, что никаких других нужных данных у нас нет. Тогда сеть целесообразно строить следующим образом. Для прогнозирования на будущую неделю надо подавать данные о продажах за последние недели, а также данные о продажах в течение

нескольких недель подряд год назад, чтобы сеть видела динамику продаж один сезон назад, когда эта динамика была похожа на настоящую за счет сезонности.

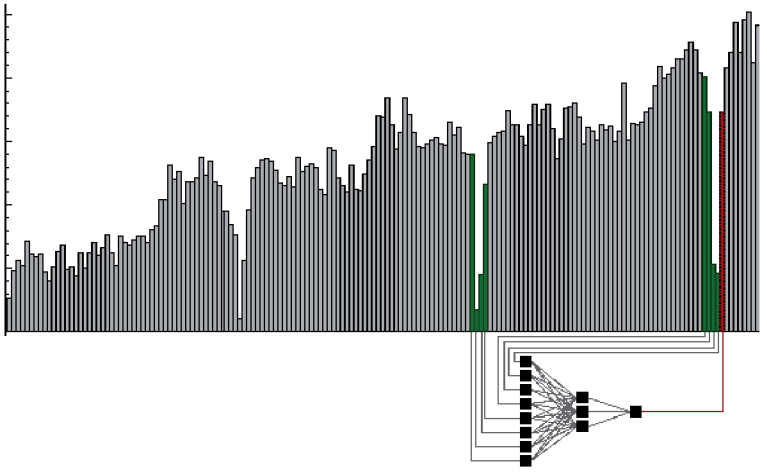


Рис. 7.3

Если входных параметров много, то рекомендуется не сбрасывать их сразу в нейронную сеть, а попытаться сначала провести предобработку данных, для того чтобы понизить их размерность или представить в правильном виде.

Предобработка данных – отдельная большая тема, которой следует уделить достаточно много времени, т.к. она является ключевым этапом в работе с нейронной сетью. В большинстве практических задач по прогнозированию продаж предобработка состоит из разных частей. Вот лишь один пример.

Пусть в предыдущем примере у нас есть не только историческая база данных о продажах продукта, которые мы прогнозируем, но и данные о его рекламе на телевидении. Эти данные могут выглядеть следующим образом(рис.7.4):

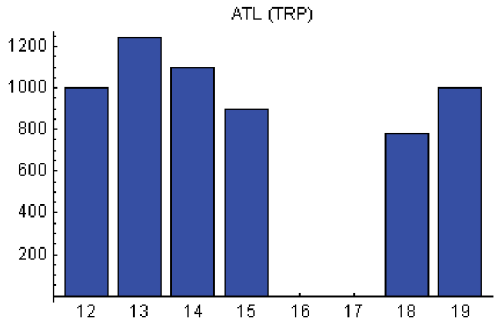



Рис. 7.4

По оси времени отложены номера недель и рекламные индексы для каждой недели. Видно, что в шестнадцатую и семнадцатую недели рекламы не было вообще. Очевидно, что неправильно в таком виде подавать в сеть (если это не рекуррентная нейронная сеть) данные о рекламе, поскольку определяет продажи не сама реклама как таковая, а образы и впечатления в сознании покупателя, которые эта реклама создает. И такая реклама имеет продолжительное действие – даже через несколько месяцев после окончания рекламы на телевидении люди будут помнить продукт и покупать его, хотя скорее всего продажи будут постепенно падать. Поэтому, подавая в сеть такие данные о рекламе, мы делаем неправильную постановку задачи и, как минимум, усложняем процесс обучения.

При использовании многослойных нейронных сетей в бизнес-прогнозировании и, в частности, прогнозировании продаж полезно также помнить о том, что нужно аккуратно делать нормировку и что для выходного нейрона лучше использовать линейную передаточную функцию. Обобщающие свойства от этого немного ухудшаются, но сеть будет намного лучше работать с данными, содержащими тренд.

### *7.4. Использование нейронных сетей с общей регрессией GRNN и GRNN-GA*

Еще одной часто используемой нейросетевой архитектурой, используемой в бизнес-прогнозировании, является нейронная сеть с общей

регрессией. Несмотря на то что принцип обучения и применения таких сетей в корне отличается от обычных персептронов, внешне сеть используется таким же образом, как и обычный персептрон. Говоря другими словами, это совместимые архитектуры в том смысле, что в работающей системе прогнозирования можно заменить работающий персептрон на сеть с общей регрессией, и все будет работать – не потребуется проводить никаких дополнительных манипуляций с данными.

Если персептрон во время обучения запоминал предъявляемые примеры, постепенно подстраивая свои внутренние параметры, то сети с общей регрессией запоминают примеры в буквальном смысле. Каждому примеру соответствует отдельный нейрон в скрытом слое сети, а затем во время применения сеть сравнивает предъявляемый пример с примерами, которые она помнит. Сеть определяет, на какие из них текущий пример похож и в какой степени, и на основе этого сравнения выдает ответ.

Отсюда следует первый недостаток такой архитектуры: если база данных о продажах или других величинах, которые мы прогнозируем, велика, то сеть станет слишком большой и будет медленно работать. С этим можно бороться при помощи предварительной кластеризации базы данных.

Второй недостаток таких сетей особенно заметен в задачах бизнес-прогнозирования они совсем не способны "продлевать" тренд. Поэтому такие сети можно использовать только в случаях, когда рынок устойчивый, либо, после декомпозиции данных, тренд прогнозировать с помощью других архитектурами нейронных сетей или любых классических методов.

## 8. Обзор методов моделирования

Когда количество и характер внешних воздействий не позволяет формализовать процедуру принятия решения, когда необходимо учесть практически непрогнозируемое влияние человеческого фактора одним (если не единственным) способом анализа системы принятия решения применяется имитационное моделирование.

В настоящем разделе приводится обзор методов имитационного, ситуационного и экспертного моделирования.

### *8.1. Имитационное моделирование*

Слишком часто в нашей жизни эксперименты ставятся на реальных системах, будь то экономика страны, отдельная организация или система управления сложным перекрестком. Лицо, принимающее решение, надеется при этом на свою проницательность, интуицию и удачу. Еще в 1969 г. известный ученый, родоначальник системной динамики Джей Форрестер отмечал, что на основе интуиции для управления сложными системами чаще выбираются неверные решения, чем верные, и это происходит потому, что в сложной системе причинно-следственные отношения ее параметров не являются простыми и ясными. В литературе имеется большое число примеров, показывающих, что люди не способны предвидеть результат их воздействий в сложных системах. Примером может служить каскадное развитие аварий в энергосистемах северо-запада США 16 августа 2003 года и в Московском регионе 25 мая 2005 года, приведших к миллиардным потерям и затронувшим миллионы людей. Невысокая эффективность управленческих решений, сделанных на основе интуиции, объясняется тем, что причины и следствия в сложных системах разнесены во времени и пространстве, поэтому человеку трудно предсказать, какие последствия вызовет то или иное решение. В тех случаях, когда для оценки принимаемых решений эксперимент с реальными системами невозможен либо слишком дорог, используется имитационное моделирование.

*Имитационное моделирование (ИМ)* — это разработка компьютерных моделей и постановка экспериментов на них. Целью моделирования в конечном счете является принятие адекватных (т. е. обоснованных, целесообразных и реализуемых) управленческих решений. Компьютерное моделирование является обязательным этапом в принятии ответственных решений во всех областях деятельности человека поскольку усложняются системы, в рамках которых человек должен действовать и которыми он

должен управлять. Знание концепций, принципов и возможностей имитационного моделирования, умение строить и использовать модели являются необходимыми требованиями к инженеру, менеджеру, бизнес-аналитику.

В английском языке для обозначения процесса моделирования используется два различных слова – *modeling* и *simulate*. Первому слову соответствует процесс проектирования, создания модели устройства или предметной области. Под вторым понятием имитацией (*simulate*) – понимают исследование (испытание, прогонку) модели. Процесс имитации невозможен без предварительного создания модели. Последующая имитация накладывает ограничения на язык и способы описания модели.

Отличительной чертой неимитационных моделей является их статичность, поэтому к ним можно отнести языки описания декларативных знаний или данных. Примерами их могут являться фактографические системы (базы данных) и модели БД (реляционная, объектная), CASE-системы проектирования ERwin и Rational Rose и соответствующие им модели и языки IDEFX1, UML.

Моделирование особенно важно именно тогда, когда система состоит из многих параллельно функционирующих во времени и взаимодействующих подсистем. Такие системы наиболее часто встречаются в жизни. Каждый человек мыслит последовательно, даже очень умный человек в конкретный момент времени обычно может думать только об одном деле. Поэтому понимание одновременного развития во времени многих влияющих друг на друга процессов является для человека трудной задачей. Имитационная модель помогает понять сложные системы, предсказать их поведение и развитие процессов в различных ситуациях и, наконец, дает возможность изменять параметры и даже структуру модели, чтобы направить эти процессы в желаемое русло. Модели позволяют оценить эффект планируемых изменений, выполнить сравнительный анализ качества возможных вариантов решений. Такое моделирование может

осуществляться в реальном времени, что позволяет использовать его результаты в различных технологиях (от оперативного управления до тренинга персонала).

Имитационное моделирование может применяться в самых различных сферах деятельности. Ниже приведен список задач, при решении которых моделирование особенно эффективно:

- проектирование и анализ производственных систем;
- оценка различных систем вооружений и требований к их материально-техническому обеспечению;
- определение требований к оборудованию и протоколам сетей связи;
- определение требований к оборудованию и программному обеспечению различных компьютерных систем;
- проектирование и анализ работы транспортных систем, например аэропортов, автомагистралей, портов и метрополитена;
- оценка проектов создания различных организаций массового обслуживания, например центров обработки заказов, заведений быстрого питания, больниц, отделений связи;
- модернизация различных процессов в деловой сфере;
- политика в системах управления запасами;
- анализ финансовых и экономических систем.

Моделирование насчитывает четыре основных направления: моделирование *динамических систем, дискретно-событийное* моделирование, *системная динамика* и *агентное* моделирование. В каждом из этих направлений развиваются свои инструментальные средства, упрощающие разработку моделей и их анализ. Данные направления (кроме агентного моделирования) базируются на концепциях и парадигмах, которые появились и были зафиксированы в инструментальных пакетах моделирования несколько десятилетий назад и с тех пор не менялись.

В современном мире информационных технологий десятилетие сравнимо с веком прогресса в традиционных технологиях, и удивительно то, что в имитационном моделировании почти без изменения и почти повсеместно

применяются идеи и решения 60-х г. прошлого века. Возникшие тогда парадигмы имитационного моделирования зафиксированы в распространяемых на рынке программных продуктах, они не используют появившихся значительно позже современных достижений информационных технологий, которые привели к революционным изменениям во многих прикладных областях. Поэтому разработка имитационных моделей в рамках традиционных парадигм остается непростой задачей. Кроме того, проблемы анализа современных реальных систем часто требуют разработки моделей, не укладывающихся в рамки одной единственной парадигмы моделирования. Например, при моделировании системы с преобладающим дискретным типом событий может потребоваться введение переменных, описывающих непрерывные характеристики среды. В парадигму блочной модели потоков данных совершенно не вписываются дискретно-событийные системы, поэтому, например, в среде Simulink с большим трудом выражаются сущности дискретно-событийных моделей: события и состояния, поведение, управляемое событиями. В рамках обеих этих парадигм невыразимы концепции активных объектов, взаимодействующих с окружением, что необходимо использовать, например, для моделирования конкуренции компаний на рынке. В системно-динамической модели часто возникает необходимость учета дискретных событий или моделирования индивидуальных свойств объектов из разнородных групп. Подобные требования, выводящие за рамки традиционных парадигм при построении моделей, требуют использования скриптовых языков, тонких и сложных средств интеграции внешних программных модулей с моделью и т.п., что существенно усложняет разработку моделей в традиционных средах.

Имитационное моделирование используется узким кругом профессионалов, которые должны иметь не только глубокие знания в той прикладной области, для которой строится модель, но и в программировании, теории вероятностей и статистике.

Имитационная модель организационно-технической системы в силу сложной структуры должна быть иерархической, что позволит применять к ней теории иерархических и мультиагентных систем.

Теоретической базой создания средств ИМ являются широко распространенные математические схемы описания динамических процессов (расширенные сети Петри, системы массового обслуживания, модели системной динамики). Новый подход к моделированию динамических процессов, к которым относятся цепочки поставок (логистика), технологические, производственные, организационные и бизнес-процессы, предлагает концепция процессов преобразования ресурсов, синтезированная на базе вышеупомянутых математических схем.

Системы имитационного моделирования (СИМ) можно разделить на два класса: универсальные и проблемно ориентированные. Проблемно ориентированные СИМ имеют одно важное преимущество – они снижают требования к конечному пользователю в области программирования, т. е. с точки зрения внедрения и применения на предприятиях, в организациях и бизнесе имеют больший шанс на выживание. К распространенным в настоящее время проблемно-ориентированным СИМ в области дискретных процессов преобразования ресурсов относятся следующие: AnyLogic, Arena, ARIS, ReThink.

Исходными данными для экспертного моделирования (СЭМ), которые имитируют процессы рассуждения человека, являются декларативные и процедурные знания, поэтому их также называют *системами, основанными на знаниях (knowledge-based system),* или *экспертными системами (ЭС).* В общем случае ЭС нельзя рассматривать как СИМ, т. к. они используют критерии, стратегии выбора правил, формализованные цели и т. д. Тем не менее, при моделировании знаний эксперта, которые представляют собой вербальное или графическое отображение системы, ее связей и закономерностей, экспертное представление аналогично имитационному.

ЭС – наиболее распространенный класс информационных систем, ориентированный на тиражирование опыта высококвалифицированных специалистов в областях, где качество принятия решений традиционно зависит от уровня экспертизы, например, в медицине, юриспруденции, геологии, экономике, военном деле, энергетике, металлургии, логистике, проектировании. ЭС эффективны лишь в специфических «экспертных» областях, в которых важен эмпирический опыт специалистов.

Существует множество различных определений экспертной системы, однако в большинстве случаев ее структура остается типовой и может включать следующие компоненты: база знаний; база данных; машина вывода; интерфейс с пользователем; модуль извлечения знаний и обучения; компонент приобретения и объяснения знаний. Первые три являются обязательными.

*База данных* (БД) хранит исходные и промежуточные данные решаемой в текущий момент задачи.

*База знаний* (БЗ) предназначена для хранения долгосрочных данных, описывающих рассматриваемую область, и правил, описывающих целесообразные преобразования данных в этой области.

Отличие БЗ от БД определяют исходя из типа хранимых знаний: в БЗ записывают правила (процедурные знания), а в БД — данные (декларативные знания). Все знания стремятся хранить единообразно, используя один язык представления знаний (ЯПЗ).

*Машина вывода,* используя исходные данные и знания, формирует такую последовательность правил, которые, будучи примененными к исходным данным, приводят к решению задачи.

*Интерфейс с пользователем* ориентирован на организацию общения со всеми категориями пользователей как в ходе решения задач, так и в ходе приобретения знаний, объяснения результатов работы.

*Модуль извлечения знаний и обучения* автоматизирует процесс наполнения ЭС знаниями, осуществляемый пользователем-экспертом, а также формирует знания на основе анализа прикладных ситуаций.

*Компонент приобретения и объяснения знаний* объясняет, как система получила решение задачи (или почему она не получила решения) и какие знания она при этом использовала, что облегчает эксперту тестирование системы и повышает доверие пользователя к полученному результату.

Моделирование в ЭС (СЭМ) представляет собой *вывод на знаниях.* Механизм вывода во многом зависит от используемого языка представления знаний и может быть логическим, нечетким, вероятностным, продукционным и т. д. К экспертному моделированию относятся методы:

- формирования, изменения, дополнения и оптимизации БЗ;
- обучения, извлечения и объяснения ЭС;
- ведения диалога и разработки интерфейса взаимодействия;
- описания предметной области на ЯПЗ;
- разработки алгоритмов и стратегий вывода;
- прогнозирования, экстраполяции и эвристического анализа;
- интеграции ЭС с другими системами.

Если при моделировании (выводе) ЭС опирается на исходные данные и на основании правил, хранимых в БЗ, получает результат, то такой вывод называется *прямым.* Если ЭС осуществляет поиск всех возможных комбинаций исходных данных, приводящих к одному (заданному) результату, то такой вывод называется *обратным.*

Большинство ЯПЗ можно представить в виде сетевой структуры (семантические сети, фреймы, сценарии, продукции), поэтому в них активно используются графовые методы поиска (в ширину, глубину и др.)

Методы имитации символической модели ЭС практически полностью совпадают с методами, используемыми в СИМ. Такое сближение имитационного и экспертного подхода приводит к идеям интеграции ЭС и СИМ.

Особо следует выделить методы комбинированного вывода, которые учитывают возможности различных ЯПЗ. Кроме того, в ЭС часто используются

методы с вызовом внешних процедур (программ) и получения из них данных. Эти методы носят название *процедур извлечения знаний*.

### 6.2. Ситуационное моделирование

Одним из перспективных направлений создания моделей принятия решений, позволяющим использовать содержательные сведения о конкретных ситуациях и отражать реальную динамику процессов, а также учитывать человеческий фактор в процессе выбора решений, является *метод ситуационного управления*. Метод оформился в начале 70-х г. XX в. трудами российских ученых В. Н. Пушкина, Д. А. Поспелова, Ю. И. Клыкова, Э. Ф. Скороходько как реакция на трудности применения точных количественных методов, в частности математического программирования народнохозяйственными объектами.

Для описания ситуаций используются *семиотические (ситуационные) языки и модели,* среди которых можно выделить следующие основные подходы:

- *дискретные ситуационные сети (ДСС);*
- *RX-коды;*
- *логика предикатов;*
- *универсальный семантический код.*

Дискретная ситуационная сеть представляет собой сложную семантическую сеть. Каждая ситуация описывается ориентированным графом (сетью), а для представления вложенности ("ситуации ситуаций") используются гиперграфы, т. е. некоторый фрагмент семантической сети, определяющий ситуацию, который может рассматриваться как одна вершина сети.

*RX*-коды представляют собой язык бинарных отношений и имеют в качестве ядерной конструкции запись следующего вида:

$$x_1 = x_2 r_2 x_3 r_3, \qquad (6.1)$$

где $x_i$ — объект или ситуация; $r_i$ — отношение.

Логика предикатов – раздел математической логики, включающий логические законы, общие для любой области объектов исследования (содержащей хоть один объект) с заданными на этих объектах предикатами (т. е. свойствами и отношениями).

Универсальный семантический код использует в качестве ядерной конструкции тройку *SAO,* которая соответствует субъекту S, совершающему действие А над объектом О.

Для реализации в ЭВМ семиотических языков используют языки представления знаний. Наиболее близким подходом к описанию семиотических конструкций является семантическая сеть. Однако сети очень медлительны при использовании операций поиска, поэтому конструкции часто представляются с помощью логики предикатов, фреймов и продукций.

Методы представления знаний в ситуационных системах и экспертных системах аналогичны. Еще больше они сблизились после активного внедрения нечеткой логики в технологии ЭС.

При ситуационном моделировании активно используются имитационные модели, следовательно, ситуационный "язык должен включать некоторые средства, присущие языкам моделирования: системное время, очереди событий, организацию квазипараллельных процессов и т. д.".

Интерпретируя определение ситуации Филипповича А. Ю. в отношении процессов преобразования ресурсов, под ситуацией будем понимать оценку совокупности характеристик объектов (образуемых на множестве элементов процесса преобразования ресурсов, средств, операций, процессов, команд управления и т. д.) и связей между ними, которые состоят из постоянных и причинно-следственных отношений, зависящих от прошедших событий и протекающих процессов.

### *6.3. Мультиагентный подход*

Для решения задачи построения моделей ЛПР на разных уровнях сложной системы целесообразно использовать теорию мультиагентных систем, новое направление развития искусственного интеллекта, информационно-

телекоммуникационных технологий и имитационного моделирования. Ниже приводится краткий обзор результатов данного направления.

Агентно-ориентированный подход уже нашел применение в таких областях, как распределенное решение сложных задач, реинжиниринг предприятий, телекоммуникации, электронный бизнес, проектирование и т. п. Важной областью применения мультиагентных технологий является моделирование. В этой области Д. А. Поспелов выделяет два класса задач. К первому классу он относит задачи распределенного управления и задачи планирования достижения целей, в которых усилия разных агентов направлены на решение общей проблемы и необходимое обеспечение эффективного способа кооперации их деятельности. В задачах второго класса агенты самостоятельно решают свои локальные задачи, используя общие, как правило, ограниченные ресурсы.

Термин «агент» происходит от латинского глагола *agere*, что означает «действовать», «двигать», «править», «управлять». Это определение правильно отражает суть и современных интеллектуальных компьютерных агентов, которые могут функционировать независимо от имени своего владельца (человека-пользователя или вычислительной системы) и решать самые разнообразные задачи по обработке информации. Для успешной работы агент должен обладать достаточными интеллектуальными способностями, чтобы в сфере своих задач замещать действия человека-владельца, должен иметь возможности взаимодействия с владельцем или пользователем для получения соответствующих заданий и передачи результатов, должен ориентироваться в среде своего существования и принимать необходимые решения.

Понятие *агент* соответствует аппаратно или программно реализованной сущности, которая способна действовать в интересах достижения целей, поставленных перед ней владельцем и (или) пользователем, и которая обладает определенными интеллектуальными способностями. В дальнейшем будем придерживаться данного определения.

Две базовые характеристики – автономность и целенаправленность - позволяют отличать интеллектуального агента (ИА) от других программных и

аппаратных объектов (модули, подпрограммы, процедуры и т. п.). Интеллектуальным агентам присущи основные свойства:

- автономность — способность функционировать без вмешательства со стороны своего владельца и осуществлять контроль собственных действий и внутреннего состояния. Автономность предполагает относительную независимость агента от окружающей среды, т. е. наличие «свободы воли», обуславливающей собственное поведение, которое должно быть обеспечено необходимыми ресурсами;
- активность — способность к организации и реализации действий;
- общительность — взаимодействие и коммуникация с другими агентами;
- реактивность — адекватное восприятие состояния среды и реакция на его изменение (соответствует рефлекторному поведению животного);
- целенаправленность, предполагающая наличие собственных источников мотивации;
- наличие базовых знаний о себе, о других агентах и об окружающей среде;
- убеждения — переменная часть базовых знаний, меняющихся во времени;
- желания — стремление к определенным состояниям;
- намерения — действия, которые планируются агентом для выполнения своих обязательств и (или) желаний;
- обязательства — задачи, которые выполняет один агент по просьбе и (или) поручению других агентов.

Иногда к этому списку добавляются другие качества:

- правдивость — неспособность к подмене истинной информации заведомо ложной;
- благожелательность — готовность к сотрудничеству с другими агентами в процессе решения собственных задач, что обычно предполагает отсутствие конфликтующих целей, поставленных перед агентами;
- альтруизм — приоритетность общих целей по сравнению с личными;
- мобильность – способность агента мигрировать по сети в поисках необходимой информации.

Наиболее известными исследовательскими центрами в области агентных систем и технологий являются университет Карнеги Мэллон (Carnegi Mallon University), Массачусетский университет (University of Massachusetts at Amherst), университет г. Болоньи (Univrsita di Bologna), ряд университетов и колледжей Великобритании (Stanford University, Manchester Metropolitan University). Занимаются этими проблемами и крупные корпорации (IBM, Microsoft, DEC, Apple, Toshiba, Hewlett Packard и др.). В нашей стране исследования по данной тематике проводятся в Исследовательском центре искусственного интеллекта Института программных систем РАН (г. Переславль-Залесский), в Институте проблем управления РАН, в Санкт-Петербургском институте информатики и автоматизации РАН, в Санкт-Петербургском государственном электротехническом университете, в Санкт-Петербургском техническом университете, в Институте проблем управления сложными системами РАН (г. Самара), в Уфимском государственном авиационно-техническом университете, в Таганрогском радиотехническом университете.

Интеллектуальная мультиагентная система (МАС) представляет собой множество интеллектуальных агентов, распределенных в сети, которые мигрируют по ней в поисках релевантных данных, знаний, процедур и кооперируются для достижения поставленных перед ними целей. В МАС множество автономных агентов действуют в интересах различных пользователей и взаимодействуют между собой в процессе решения определенных задач. Примерами таких задач являются: управление информационными потоками и сетями, управление воздушным движением, поиск информации в сети Интернет, электронная коммерция, обучение, электронные библиотеки, коллективное принятие многокритериальных управленческих решений и др.

Основными направлениями научного поиска являются теории агентов, которые рассматривают математические методы, и формализмы абстрактного представления структуры и свойств агентов, и способы построения рассуждений в таких формальных системах; методы коллективного поведения агентов; архитектуры агентов и МАС; методы, языки и средства

коммуникации агентов; языки программирования агентов; методы и средства автоматизированного проектирования МАС; методы и средства обеспечения мобильности агентов.

Главная черта МАС, отличающая их от других интеллектуальных систем, — взаимодействие между агентами. Взаимодействие означает установление двусторонних и многосторонних динамических отношений между субъектами. Оно является не только следствием деятельности агентов, но и необходимым условием формирования виртуальных сообществ. К базовым видам взаимодействия между агентами относятся: *кооперация* (сотрудничество); *конкуренция* (конфронтация, конфликт); *компромисс* (учет интересов других агентов); *конформизм* (отказ от своих интересов в пользу других); *уклонение от взаимодействия.*

Взаимодействие агентов обусловлено целым рядом причин, важнейшими среди которых являются следующие.

*Совместимость целей (общая цель).* Эта причина обычно порождает взаимодействие по типу кооперации. Несовместимость целей или убеждений обычно порождает конфликты, позитивная роль которых заключается в стимулировании процессов развития.

*Отношение к ресурсам.* Ресурсами будем называть любые средства, используемые для достижения агентами своих целей. Ограниченность ресурсов, которые используются многими агентами, обычно порождает конфликты. Одним из самых простых и эффективных способов разрешения подобных конфликтов является право сильного — сильный агент отбирает ресурсы у слабых.

*Необходимость привлечения недостающего опыта.* Каждый агент обладает ограниченным набором знаний, необходимых ему для реализации собственных и общих целей. В связи с этим агенту приходится взаимодействовать с другими агентами. В таком случае возможны различные ситуации: а) агент способен выполнить задачу самостоятельно; б) агент может обойтись без посторонней помощи, но кооперация позволит решить задачу более эффективным способом; в) агент не способен решить задачу в одиночку. В зависимости от ситуации агенты

выбирают тип взаимодействия и могут проявлять разную степень заинтересованности в сотрудничестве.

*Взаимные обязательства.* Обязательства являются одним из инструментов, позволяющих упорядочить хаотические взаимодействия агентов. Они позволяют предвидеть поведение других агентов, прогнозировать будущее и планировать собственные действия. Формальное представление целей, обязательств, желаний и намерений, а также всех остальных характеристик составляет основу ментальной модели интеллектуального агента, которая обеспечивает его мотивированное поведение в автономном режиме.

Перечисленные причины в различных сочетаниях могут приводить к разным формам взаимодействия между агентами, например:

- простое сотрудничество, которое предполагает интеграцию опыта отдельных агентов (распределение задач, обмен знаниями и т. п.) без специальных мер по координации их действий;
- координируемое сотрудничество, когда агенты вынуждены согласовывать свои действия для того, чтобы эффективно использовать ресурсы и собственный опыт;
- непродуктивное сотрудничество, когда агенты совместно используют ресурсы или решают общую проблему, не обмениваясь опытом и мешая друг другу.

Коллективное поведение агентов в МАС предполагает кооперацию агентов при коллективном решении задач. В процессе работы мультиагентной системы агент может обращаться за помощью к другим агентам, если не в состоянии решить поставленную перед ним задачу самостоятельно. При этом агенты могут строить планы совместных действий, не только полагаясь на свои возможности, но и анализируя планы и намерения других членов коллектива. Моделирование коллективного поведения необходимо также в случаях, когда агенты для решения своих задач используют общий ограниченный ресурс. Каждый агент вынужден учитывать наличие других агентов, а выбор стратегии действий одного агента обычно зависит от поведения остальных.

Известные подходы проектирования агентно-ориентированных систем можно разделить на две группы:

- базирующиеся на объектно-ориентированных методах и технологиях с использованием соответствующих расширений;
- использующие традиционные методы инженерии знаний.

В методологиях первой группы разрабатываются расширения объектно-ориентированных методов и технологий для проектирования агентно-ориентированных систем. Существует ряд CASE-средств, поддерживающих объектно-ориентированные методы разработки ИС, среди которых наиболее известными являются All Fusion фирмы Computer Associate и Rational Rose фирмы IBM, процесс проектирования в которых основывается на языке объектно-ориентированного проектирования UML. Современное средство имитационного моделирования AnyLogic, поддерживающее агентный подход, использует расширение языка UML-RT; подробнее данный инструментарий рассматривается в разделе 6.5.

Вторая группа методологий строится на расширении традиционных методов инженерии знаний. Эти методологии обеспечивают формальные и композиционные языки моделирования для верификации структуры системы и функций. Данные подходы применимы к моделированию знаний и информационно-ориентированных агентов.

Заканчивая обзор методов моделирования, необходимо отметить то, что специально разработанных СДМС в предметной области процессов преобразования ресурсов не существует, поэтому актуальным является разработка СДМС, поддерживающая следующие функциональные возможности:

- описание ситуационной модели в виде дискретной ситуационной сети, как наиболее соответствующей процессам преобразования ресурсов;
- декомпозицию ситуационной модели;
- представление информации и моделей с использованием когнитивной графики;
- описание моделей ЛПР в виде интеллектуальных агентов, обрабатывающих информацию, диагностирующих ситуации,

вырабатывающих решения (работающих со знаниями), действующих в соответствии с найденным решением и своей моделью поведения (обладающих моделью поведения), участвующих в обмене сообщениями с другими агентами;

- вывод на знаниях;
- имитационное дискретно-событийное моделирование.

В рамках данной работы ставится и решается задача интеграции следующих математических аппаратов: имитационного моделирования (дискретных процессов преобразования ресурсов), экспертных систем, ситуационного и мультиагентного моделирования.

## Библиографический список

Аксенов К. А. Динамическое моделирование мультиагентных процессов преобразования ресурсов: монография / К. А. Аксенов, Н. В. Гончарова. Екатеринбург : ГОУ ВПО УГТУ-УПИ, 2006. 311 с.

Барский А. Б. Нейронные сети: распознавание, управление, принятие решений /А. Б. Барский. М. : Финансы и статистика, 2004. 176 с.

Дуда Р. Распознавание образов и анализ сцен / Р. Дуда, П. Харт. М.: Мир, 1976. 507 с.

Рутковская Д. / Нейронные сети, генетические алгоритмы и нечеткие системы / Д. Рутковская, М. Пилиньский, Л. Рутковский. М. : Горячая линия – Телеком, 2006. 452 с.

Тархов Д. А. Нейронные сети. Модели и алгоритмы / Д. А. Тайхов М. : Радиотехника, 2005. 256 с.

Хайкин С. Нейронные сети / С. Хайкин. М. : Вильямс, 2006. 1103 с.